

No. 2014-1261

**UNITED STATES COURT OF APPEALS
FOR THE FEDERAL CIRCUIT**

FUZZYSHARP TECHNOLOGIES INC.,

Plaintiff-Appellant,

v.

INTEL CORPORATION

Defendant-Appellee.

Appeal from the United States District Court for the Northern District of
California in Civil Action No. 12-CV-04413 YGR District Judge Yvonne G.
Rogers

CORRECTED
OPENING BRIEF AND ADDENDUM OF
PLAINTIFF-APPELLANT FUZZYSHARP TECHNOLOGIES INC.

Date: August 20, 2014

David Fink
Fink & Johnson
7519 Apache Plume
Houston, TX 77071
(713) 729-4991
Attorney for Plaintiff-Appellant
FuzzySharp Technologies Inc.

CERTIFICATE OF INTEREST

Counsel for Plaintiff-Appellant certifies the following:

1. I represent FuzzySharp Technologies Inc.
2. The name of the real party in interest (if the party named in the caption is not the real party in interest) represented: Not Applicable.
3. All parent corporations and any publicly held companies that own 10 percent or more of the party or amicus curiae represented: None.
4. The names of all law firms and the partners and associates that appeared for the party or amicus now represented in trial court or agency or expected to appear in this court are:

Fink & Johnson

David Fink

Date: August 20, 2014

Respectfully submitted,

By: /s/ David Fink

David Fink

Fink & Johnson

Attorney for Plaintiff-Appellant

TABLE OF CONTENTS	Page
STATEMENT OF RELATED CASES	1
STATEMENT OF JURISDICTIONAL	2
STATEMENT OF ISSUES	3
STATEMENT OF THE CASE	4
STATEMENT OF THE FACTS.....	5
A. This Appeal is directed to only Claim 67 of the ‘047 Patent	5
B. Claim 67 of the ‘047 Patent	8
C. Explanation of Claim 67 in Plain English	9
D. Claim Construction of the Terms in Claim 67	13
E. A Close Look at Claim 67	14
SUMMARY OF THE ARGUMENT.....	18
A. LEGAL STANDARD	19
B. THE DISTRICT COURT FAILED TO CONSIDER CLAIM 67 DIRECTLY	22
C. THE DISTRICT COURT REACHED ITS DECISION OF UNPATENTABILITY PRIOR TO THE CLAIM CONSTRUCTION	23
D. CLAIM 67 DOES NOT PRE-EMPT THE CALCULATION OF HIDDEN SURFACES OR SUB-ELEMENTS	24

E. CLAIM 67 IS NOT IN CONFLICT WITH <i>MAYO</i>	24
F. FLOOK DOES NOT APPLY TO CLAIM 67	25
G. AWARDING COSTS TO INTEL IS INAPPROPRIATE IN VIEW OF THE ABUSE OF DISCRETION OF THE DISTRICT COURT	25
VI. CONCLUSION	26
ADDENDUM	29
Order, Dated November 6, 2013 (DKT 74), CONSTRUING CLAIM TERMS IN DISPUTE AND GRANTING SUMMARY JUDGMENT IN FAVOR OF DEFENDANT INTEL CORPORATION. (“Summary Judgment”) A1-A25	
Order, Dated December 11, 2013 (DKT 83), FINAL JUDGMENT. (“Final Judgment”) A26-A28	
U.S. Patent No. 6,618,047	A29-A68
U.S. Patent No. 6,172,679	A69-A107

TABLE OF AUTHORITIES

Pages

FEDERAL CASES

<i>Diamond v. Diehr</i> , 450 U.S. 175 (1981)	20, 24
<i>Mayo Collaborative Serv. v. Prometheus Labs., Inc.</i> , 132 S. Ct. 1289, 1301 (2012)	20, 24
<i>Gottschalk v. Benson</i> , 409 U.S. 63, 67 (1972)	20
<i>CLS Bank Int’l v. Alice Corp. Ltd.</i> , 717 F. 3d 1269, 1282 (Fed. Cir. 2013)	20
<i>Ultramercial, Inc. v. Hulu, LLC</i> , 722 F 3d 1335 (Fed. Cir. 2013)	21, 22
<i>Parker v. Flook</i> , 437 U.S. 584 (1978)	21, 25

FEDERAL STATUTES AND RULES

28 U.S.C. § 1295(a)(1)	2
Fed. R. App. P.4(a)(4)(B)(i)	2
28 U.S.C. §§ 1338(a) and 1400(b)	2
28 U.S.C. § 2107	2
35 U.S.C. § 101	3, 4, 18, 19, 21, 24, 25

STATEMENT OF RELATED CASES

There have been no other appeals before this or any other appellate court stemming from the civil action giving rise to this appeal.

Each of the two patents on appeal has a pending *Inter Partes* case in the U.S. Patent and Trademark Office: Case IPR2014-00001 for U.S. Patent No. 6,618,047 (“‘047 Patent”, A29-A68) and Case IPR2014-400002 for U.S. Patent No. 6,172,679 (“‘679 Patent”, A69-A107). The Decisions of the *Inter Partes* cases could impact the validity of one or more claims in the two patents before this Court.

I. STATEMENT OF JURISDICTION

Plaintiff-Appellant, FuzzySharp Technologies Inc. (“FuzzySharp”), brings this Appeal from the United States District Court of the Northern District of California (the “District Court”) to the United States Court of Appeals for the Federal Circuit in accordance with 28 U.S.C. § 1295(a)(1). *See* Fed. R. App. P.4(a)(4)(B)(i).

This is a case arising under the United States Patent Laws and Jurisdiction is in accordance with and pursuant to 28 U.S.C. §§ 1338(a) and 1400(b). FuzzySharp is appealing the orders of the District Court entered on November 6, 2013 Construing Claim Terms and Granting Summary Judgment in Favor of Defendant Intel Corporation (“Summary Judgment”), and the subsequent final judgment (“Final Judgment”) entered on January 23, 2014 against FuzzySharp. These Orders are the final Orders disposing of all of FuzzySharp’s asserted claims before the District Court.

This Appeal has been timely taken timely pursuant to 28 U.S.C. § 2107 based on the Order by the District Court dated January 23, 2014 (DKT 90) extending the time to file the Notice of Appeal until January 27, 2014.

II. STATEMENT OF THE ISSUES

1. Whether the District Court abused its discretion by granting of the Summary Judgment for invalidity under 35 U.S.C. § 101, and corresponding entry of judgment of invalidity was erroneous.
2. Whether the District Court abused its discretion by failed to address the specific terms of independent claims based on the claim construction made by the District Court.
3. Whether the District Court abused its discretion by improperly analyzed the groups of claims using generalized steps instead of addressing each claim separately using the claim construction determined by the District Court.
4. Whether the District Court abused its discretion by failing to recognize that the claimed subject matter in the '047 Patent and the '679 Patent constitute improvements in the prior art methods as stated in the respective asserted claims.
5. Whether the District Court abused its discretion by converting Intel's Motion on the Pleadings for Judgment on March 15, 2013, into a Motion for Summary Judgment on April 29, 2013 in anticipation of the Court engaging in a *pro forma* claim construction to enable the

Court's Summary Judgment and Final Judgment many months later without actually considering the claim construction for all of the asserted claims.

6. The District Court abused its discretion by awarding costs to Intel.

III. STATEMENT OF THE CASE

FuzzySharp filed a lawsuit against Intel for infringing the '047 Patent and the '679 Patent on August 22, 2012. Intel filed a Motion on the Pleadings for Judgment on March 15, 2013, and after being opposed, the Court converted the Intel's Motion into a Motion for Summary Judgment on April 29, 2013. After FuzzySharp and Intel filed additional Briefs, the Court granted a Stipulation to Extend Time to Exchange Patent Rule 4-2 Disclosures on May 31, 2013.

Thereafter, the parties agreed on most terms in a Final Joint Claim Construction filed September 9, 2013. The held a Markman Hearing on October 9, 2013 and a week later, and then had a Hearing on the Motion for Summary Judgment. The District Court granted the Motion for Judgment for Patent Ineligibility of the asserted Claims under Section 101 to Intel on November 11, 2013 (A1-A25). The District Court entered the Final Judgment on December 11, 2013 (A26-A28).

IV. STATEMENT OF THE FACTS

A. This Appeal is directed to only Claim 67 of the ‘047 Patent

The invention in Claim 67 of the ‘047 Patent is the only claim that is the subject of this Appeal. All of the other asserted claims are not being appealed, but not as an agreement that the District Court has correctly applied the facts and the law.

The claimed invention in Claim 67 discloses an improvement in rendering a 3D computer graphics image in digital format into a 2D image format suitable for a computer display or the like. Although Claim 67 is not in a classic Jepson claim format, but does admit the prior art and identifies the improvement.

A 3D computer graphics image of a scene can include objects partly or completely hidden from the point of view used to render the 2D image. The graphics image is in digital format and stored in a computer memory. In the prior art, the entire 3D computer image would be processed including shading and coloring, but before the final 2D format was created for a computer display, the computer processor perform an analysis on the entire processed computer image to eliminate hidden portions of the scene to avoid having the computer display look weird such as having a tree coming out of a person’s body.

The earliest and probably the most reliable test for hidden portions was

done with projecting imaginary lines along a z-axis into a digital scene and having the computer determine in digital space if a pixel is hidden so that pixel would not be rendered on a computer display. The computer memory kept track of all of the pixels not to be displayed. This z-buffer process remains today as a highly reliable process for eliminating hidden portions from a 3D scene to be rendered in 2D on a computer display, and is often used as the final test of visibility before a processed scene is ready for a computer display.

The inventor of the '047 Patent, Dr. Hong Lip Lim, recognized while in graduate school that there was a great need to speed up the processing of a 3D scene for rendering in 2D on a computer display. This was important back in 1991, the date of the earliest filing in Australia, when computer were relatively extremely slow as compared to modern computers, and it is still important despite the huge advances in computer processing. The simple goal claimed in Claim 67 is to eliminate large chunks of objects that were "hidden" from the viewpoint of the observer of the scene. This approach would substantially reduce a detailed process for identifying hidden pixel in a final test for hidden portions such as the z-buffer process; however, the step of eliminating large chunks of hidden objects had to be done with care to avoid eliminating visible pixels so that the final rendering did not look strange such as a hole in someone's stomach.

The approach taken by Dr. Lim was to create a “bounding volume” around the portion of an object to be tested for visibility. The District Court accepted the claim construction for a “bounding volume” as the smallest imaginary right quadrangular prism just enclosing the 3-D object’s surfaces and whose edges are parallel to the axes of the group coordinate system that approximates the 3D object’s surfaces”. See the Summary Judgment, A21, footnote 5 for a shortened version.

How a scene “looks” is actually a view of all of the surfaces available to the viewer at the viewpoint. Thus, the bounding volume is used to enclose a surface to be “tested” for visibility.

The use of a bounding volume rather than the surface itself, provides a cautious approach against eliminating a surface as “hidden’ when only a portion of that surface may be hidden. This novel approach is “fuzzy” on the side of caution by avoiding eliminating “visible’ or partially “visible” surfaces, but “sharp” in quickly eliminating completely hidden surfaces. Hence, the name of the Plaintiff is “FuzzySharp”, thereby acknowledging the unique approach to the claimed process.

The elimination of hidden surfaces prior to other processing by the computer, particularly the final “test” of visibility can greatly speeds up rendering a 2D image. Thus, the claimed invention has enormous value to the relevant industry.

B. Claim 67 of the ‘047 Patent

Claim 67 of the ‘047 Patent is as follows:

A method in a 3-D Computer graphics for processing the visibility of 3-D surfaces before a subsequent step of visibility computations, said method comprising:

for each selected 3-D surface or sub-element of a 3-D surface, identifying grid cells on a projection plane which are under or related to a projection associated with the bounding volume of said selected 3-D surface or sub-element, said projection and said subsequent step of visibility computations are from the perspective and said subsequent step of visibility computations are from the perspective of a same viewpoint;

for each of said grid cells, accessing the corresponding z-buffer element;

and

computing the visibility of the part of the bounding volume that

projects onto said each of said grid cells by comparing the depth-related data stored in said corresponding z-buffer element with the depth-related value associated with said part.

Terms in Claim 67 have constructions in the Final Joint Claim Construction () and approved by the District Court. Before setting forth the claim construction, a simple explanation of Claim 67 might be helpful in visualizing the claim process and appreciating the remarkable nature of the claimed invention.

C. Explanation of Claim 67 in Plain English

The following is an introduction to the Claim 67 from a simplified case to explain the general idea of the claim. Imagine looking perpendicular to a glass window at a 3D scene. That is your viewpoint. If you view the scene using only one eye, then you will be looking at a 2D image of the 3D scene. Now, image that the scene you are viewing is a picture projected onto the glass window. The “window” corresponds to the “projection plane” in the claim. If you visualize an imaginary lattice structure like graph paper on the projection plane, then this is the grid cells in the claim. The image on the grid cells is the projection of all of the surfaces in the scene facing you and not hidden by any other surfaces. That is the nature of the real world as oppose to viewing a scene with an x-ray machine revealing everything within its range, whether in front or in back. Thus, your eye

has no problem in identifying what is visible in the real world, and you need not make any decision as to what is hidden, or occluded in perceiving a scene.

When the computer looks at the same scene, the computer is “looking” at a digital representation of the scene stored in the computer memory, and this means that the computer creates a digital representation of everything in the scene in front, in back and in between. For example, a dog standing in a house would not be “hidden” from the computer representation. In fact, everything in the house forms part of the computer representation within the “digital space” of the computer.

Care must be taken to use the computer representation to create digital information to be used to create an image on a computer display. One well accepted prior art approach is to test each and every part of the computer display with an imaginary line to determine which pixels were in front, or stating it in another way, which pixels are occluded. It is evident that this is an absolute test to avoid having a computer display with strange images such as having a dog’s head sticking out of the front wall of the house.

Testing each and every pixel requires a huge number of computer steps and can be time consuming, but the results are extremely reliable, and widely accepted.

Claim 67 allows blocks of pixels to be eliminated from the final detailed

process for checking for visible and hidden pixels, thereby reducing the computation time for rendering a scene for a computer display.

As pointed out, the computer representation of a scene shows everything in the scene including the surfaces which would be hidden if you visually observed the scene through a window. For a person viewing the scene through the projection plane having the grid lines, the digital representation of the scene results on projection plane were simple because only the visible surfaces appeared. In contract, for the computer representation, all surfaces facing the projection plane can appear because no digital surface blocks another from being visible in a digital sense. Blocking or occlusion does not occur until the rendering stage for the computer display.

Imagine that there is some surface in the 3D scene that you want to “test” to determine if it will be visible in a computer display, from your viewpoint. We create an imaginary “box”, a bounding volume, around the surface and look at the projection of the “box” on the projection plane. The “box” projects on certain grid blocks. Then, the computer looks to see if any other surfaces have projections on grid blocks associated with the “box”. For each of the grid blocks having a projection related to the “box’ and to another surface, the computer looks into its memory, the z-buffer, and compares the distances. If, for any of the projection of

the “box” on the projection plane, there is a surface or surface sub-element a shorter distance away, then the “box” would not be occluded. If, however, there is a distance to a surface or surface sub-element a shorter distance away for each and every projection of the box on the projection plane, then the box is hidden. Hence, the surface in the “box” need not be include in subsequent calculations, thereby achieving a reduction of calculations. The reduction in calculations depends on the complexity of the scene and there is no doubt that some clever person could create a scene in which there is little or no reduction of calculations; however, the wide spread use of the invention as evidenced by the many licensed companies confirms the value of the invention.

The use of a “box” provides a huge advantage to allow a possible error to be on the side of including the surface, rather than risk excluding the surface, thereby creating a potentially serious distortion of the computer display as compared to the actually scene. Including a surface which may actually be occluded or is mostly occluded is not a problem because a subsequent step of visibility calculation using known methods as indicated by the claim can correct any error by treating the surface as being occluded.

D. Claim Construction of the Terms in Claim 67

The District Court accepted the claim construction the parties agreed to in the Final Joint claim Construction and Prehearing Statement (DKT 64), A108-A118, and pointed out that the only dispute between the parties was whether the claim terms require 3-part categorization (*i.e.*, totally hidden vs. remaining, totally visible v. remaining, or totally v.s. totally hidden). The disputed terms relate only to claim 1 of the '047 Patent, along with claims depending on that claim, and claim 1 of the '679 Patent, along with claims depending on that claim.

This appeal is for Claim 67 which is an independent claim so no issue existed between the parties, and no change in the proposed claim construction was stated or suggested by the District Court; however, the claim construction of terms in other claims can impact the understanding of Claim 67.

The following are the claim constructions of the terms in Claim 67:

- (a) “3-D computer graphics” means “the field of three-dimensional computer graphics” see A113;
- (b) “3D surfaces” or “sub-elements” means surfaces (*i.e.*, the exterior) of 3D objects that are part of a scene. See A109;
- (c) “visibility computations” means “prior art types of rendering computations that determine which pixels of 3D surface(s) are visible,

which include z-buffering and other visibility tests performed during rendering”. See A110 ;

- (d) “grid cells” means “imaginary lattice structure”. See A111;
- (e) “projection plane” means a “representation(s) of a 3-D object’s surfaces upon an imaginary plane”. See A110;
- (f) “projection(s)” means “representation(s) of a 3-D object’s surfaces upon an imaginary plane”. See A110;
- (g) “bounding volume” means “the smallest imaginary right quadrangular prism just enclosing the 3-D object’s surfaces and whose edges are parallel to the axes of the group coordinate system that approximates the 3D object’s surfaces”. See A109
- (h) “z-buffer” means “a data structure in memory that is used to store depth data”. See A114
- (i) “depth related” means “distance from a viewpoint along th z-axis”.
See A114

E. A Close Look at Claim 67

The following is a look at the steps of Claim 67 with reference to the previous simplified description and with some comments:

Claim 67 A method in a 3-D Computer graphics for processing the visibility of 3-D surfaces before a subsequent step of visibility computations, THIS PLACES THE METHOD WITHIN THE OPERATIONS OF A COMPUTER, AND INDICATES SUBSEQUENT STEPS SO THAT THE CLAIMED INVENTION IS AN IMPROVEMENT OVER KNOWN PROCESS.

said method comprising:

for each selected 3-D surface or sub-element of a 3-D surface, identifying grid cells on a projection plane which are under or related to a projection associated with the bounding volume of said selected 3-D surface or sub-element, said projection and said subsequent step of visibility computations are from the perspective and said subsequent step of visibility computations are from the perspective of a same viewpoint; THE INVENTION IS CONCERNED WITH “SURFACES” BECAUSE ULTIMATELY, THE COMPUTER DISPLAY SHOWS SURFACES. THE PROJECTION PLANE IS PERPENDICULAR TO THE VIEWPOINT WHICH IS ALONG THE Z-AXIS, IN TERMS OF THE COORDINATE SYSTEM FOR THE VIEWPOINT. THE COMPUTER PROGRAM SELECTS A SURFACE OR PART OF A SURFACE, A SUB-ELEMENT, TO BE

EVALUATED FOR VISIBILITY. THE SURFACE OR SUB-ELEMENT IS WITHIN AN IMAGINARY BOUNDING VOLUME AND THE ACTUAL TEST OF VISIBILITY IS MADE ON THE BOUNDING VOLUME, THEREBY MAKING THE EVALUATION CAUTIOUS TO AVOID DISCARDING A SURFACE OR SUB-ELEMENT ESSENTIAL FOR A GOOD LOOKING FINAL COMPUTER DISPLAY BECAUSE EVERY SINGLE PIXEL IN THE SURFACE OR SUB-ELEMENT MUST BE HIDDEN IF THE BOUNDING VOLUME IS HIDDEN. THE BOUNDING VOLUME HAS A PROJECTION ON THE PROJECTION PLANE AND THE COMPUTER GENERATES IMAGINARY GRID CELLS. THE PROJECTION OF THE BOUNDING VOLUME ON THE GRID CELLS ARE IDENTIFIED BY THE COMPUTER AND THE COMPUTER NOTES OF ANY OTHER SURFACES ALSO HAS PROJECTIONS ON ANY OF THE SAME GRID CELLS AS THE BOUNDING VOLUME. FOR ANY GRID CELL HAVING BOTH PROJECTIONS FROM THE BOUNDING VOLUME AND SOME OTHER SURFACE, IT MEANS THAT THESE RESPECTIVE PORTIONS ARE IN LINE WITH EACH OTHER, BUT IT IS NOT

OBVIOUS AS TO WHICH IS OCCLUDED.

for each of said grid cells, accessing the corresponding z-buffer element;

and

computing the visibility of the part of the bounding volume that

projects onto said each of said grid cells by comparing the depth-

related data stored in said corresponding z-buffer element with the

depth- related value associated with said part. THE COMPUTER Z-

BUFFER HAS ALL OF THE DEPTH RELATED DATA. FOR

EACH GRID CELL HAVING BOTH BOUNDING VOLUME AND

OTHER SURFACE PROJECTIONS, A COMPARISON OF THE

DEPTH RELATED DATA IS MADE. IF ALL OF THE

BOUNDING VOLUME DEPTH RELATED DATA IS

DETERMINED TO BE GREATER THAN THE OTHER SURFACE

RELATED DEPTH DATA, THEN THE BOUNDING VOLUME IS

OCCLUDED AND EVERY SURFACE WITH THE BOUNDING

VOLUME IN OCCLUDED.

Although Claim 68 is not amongst the asserted claims, claim 68 shows the value of the subsequent use of Claim 67 to reduce the computation for visibility.

This, of course, was evident from the numerous mentions of this important

achievement in the specification and drawings as filed.

V. SUMMARY OF THE ARGUMENT

The District Court abused its discretion by failing to apply the specific limitations of asserted Claim 67 in its analysis of patentability under 35 U.S. C. § 101. Instead, the Court broadly described all of the asserted claims of the ‘047 Patent as having three components:

1. a mathematical manipulation of data;
2. a reducing step; and
3. an indication of where these steps fit in the visibility computations. See Addendum, Page 21 of the Summary Judgment. Thus, the Court failed to consider the specific limitation such as the “bounding volume” used to cautiously determine occluded surfaces and thereby save computation steps while assuring that the final computer display would not look weird.

The District Court abused its discretion by failing to apply the law to Claim 67. Instead, the Court recited standard cases and made general references to the specification, not the claims, or particularly Claim 67.

The District Court abused its discretion by failing to apply the limitation of the “bounding volume” in Claim 67.

The District Court abused its discretion by failing to note:

- a. The Claim 67 is a process, thereby falling into one of the potentially patentable subject matters. Hence, Claim 67 is *prima facie* patentable.
- b. The Claim 67 cannot be carried out in a person's head. A computer is required. It could be a general computer.
- c. The Claim 67 is an improvement in the field of processing digital data related to a 3D scene to form a 2D computer display so that extensive details are not needed for each and every step within the known technology.
- d. As an improvement in a known field, the Claim 67 does not preclude generating a 2D computer display from a 3D scene.
- e. The Claim 67 includes a novel step of using a bounding volume to reduce the computational steps without adversely impacting the final 2D computer display.

A. LEGAL STANDARD

The issue that was before the District Court is whether the claimed invention as claimed in Claim 67 is patentable under 35 U.S.C. § 101. Section 101 specifies four statutory classes of patentable subject matter: “process, machine, manufacture or composition of matter”.

The Supreme Court decide that the scope of these classes of inventions was vague and some limitations were needed. Hence, “laws of nature, physical phenomena, and abstract ideas” were excluded. *Diamond v. Chakrabarty*, 447 U.S. 303, 309 (1980). The Supreme Court expressed “concern that patent law not inhibit further discovery by improperly tying up use of laws of nature, natural phenomena, and abstract ideas.” *Mayo Collaborative Serv. v. Prometheus Labs., Inc.* 132 S. Ct. 1289, 1301 (2012). The Supreme Court has also stated, “A principle, in the abstract, is a fundamental truth ...[which] cannot be patented.” *Gottschalk v. Benson*, 409 U.S. 63, 67 (1972) (internal citations and quotations omitted”). “Phenomena of nature, though just discovered, mental processes, and abstract intellectual concepts are not patentable, as they are tools of scientific and technological work.” *Id.* Included within the exception for abstract ideas are such concepts as mathematical formulae. *id.*

It appears that the Supreme Court thought that people might believe that they “invented” a law of nature, or “invented” an idea, or the like so these discoveries were explicitly excluded. But confusion has persisted.

Recently, this Court raise the question as to whether a patent claim can “pose any risk of preempting an abstract idea.” *CLS Bank Int’l v. Alice Corp. Ltd.*, 717 F. 3d 1269, 1282 (Fed. Cir. 2013). Instead of being a limitation, this is an

exclusion clause by the Court from a possible rejection because the Court clearly allows the use of an abstract idea such as a mathematical formula in a patent claim as long as the formula is a “tool” in the operation or implementation of the claim, thereby limiting the scope of the abstract idea against future use. In simple terms, the use of a mathematical formula does not invalidate a patent claim any more than a claim step of moving an object from one place to another. Obviously, the step of “moving” does not preclude any one from “moving” anything else or incorporating a “moving” step into some other invention.

In *Parker v. Flook*, 437 U.S. 584 (1978), the Court stressed the importance of looking for an “inventive concept” in the patent claim.

This Court recently held that “any attack on an issued patent based on a challenge to the eligibility of the subject matter must be proven by clear and convincing evidence.” *Ultramercial, Inc. v. Hulu, LLC*, 722 F.3d 1335 (Fed. Cir. 2013). The *Ultramercial* Court found claims concerning a method for distributing media products over the Internet to be patent-eligible.

The common denominator of all relevant case law to Section 101 as illustrated by *CLS Bank Int’l* is that the District Court must determine if a patent claim is patentable according to Section 101 based on the specific language of the patent claim, and not a generalized view of the claim of the claim, even if there is

no claim construction.

B. THE DISTRICT COURT FAILED TO CONSIDER CLAIM 67 DIRECTLY

The District Court made its conclusion that Claim 67 was unpatentable without actually addressing the claim language. Instead, the Court lumped all of the asserted claims of the '047 Patent in a general statement of operations, and then asserted that the claims, not Claim 67 particularly, lack a meaningful limitation. See p. 22 of the Summary Judgment.

The District Court reasoned that the Claim 67 indicates that the application of the “formula” is extremely broad, based on the specification mentioning varies area suitable for applying the invention. See p. 22 of the Summary Judgment. The District Court ignored the obvious fact that the two patents-in-suit have 100 patent claims and instead, the District Court reached its conclusion apparently assuming that all the possible applications of the invention were incorporated into a single asserted claim, or each and every asserted claim.

This Court recently held that “any attack on an issued patent based on a challenge to the eligibility of the subject matter must be proven by clear and convincing evidence.” *Ultramercial, v. Hulu, LLC*, 722 F. 3d 1335, 1342 (Fed. Cir. 2013).

The District Court failed to prove by clear and convincing evidence that Claim 67 is unpatentable because the District Court did not address Claim 67 directly.

C. THE DISTRICT COURT REACHED ITS DECISION OF UNPATENTABILITY PRIOR TO THE CLAIM CONSTRUCTION

Intel filed a Motion for Judgment on the Pleadings Dismissing FuzzySharp's Claims for Lack of Patent-Eligibility on March 15, 2013 (DKT 28) before any claim construction was made. The logical response for the District Court after the Motion was opposed, was to deny the Motion. The primary argument in the Motion was that the previous District Court had held most of the claims patent ineligible; however, this ignored the fact that this Court remanded the prior appeal with a different Defendant because no claim construction had been made by that District Court.

Instead of denying the Motion, the District Court merely placed the Motion on hold to complete a claim construction, thereby sending a clear message that the District Court would have a Markman Hearing to avoid the remand of the previous case. Hence, the District Court abused its discretion by not applying the claim construction to Claim 67 to reach its conclusion.

D. CLAIM 67 DOES NOT PRE-EMPT THE CALCULATION OF HIDDEN SURFACES OR SUB-ELEMENTS

Nothing in the analysis of the District Court states or suggests that the Claim 67 pre-empts the calculation of hidden surfaces or sub-elements. Thus, *Diamond v. Diehr*, 450 U.S. 175 (1981) does not preclude patentability. The Diehr Court held that the patentees did not “seek to pre-empt the use of the equation. Rather, they seek only to foreclose from others the use of that equation *in conjunction with all of the other steps* in their claimed process.”

E. CLAIM 67 IS NOT IN CONFLICT WITH *MAYO*

The Supreme Court recently held that claims were too abstract, and not patentable under section 101, where “the claimed processes (apart from the natural laws themselves) involve well-understood, routine, conventional activity previously engaged in by researchers in the field...” *Mayo Collaborative Serv. v. Prometheus Labs., Inc.*, 132 S. Ct. 1289, 1301 (2012).

The District Court never asserted or suggested that Claim 67 was abstract because it involved well known, routine, conventional activity engaged in by other people in the field.

F. FLOOK DOES NOT APPLY TO CLAIM 67

In *Parker v. Flook*, 437 U.S. 584 (1978), the Court stressed the importance of looking for an “inventive concept”. As pointed out, the District Court applied its reasoning and conclusion that all of the asserted claims of the ‘047 Patent are unpatentable under 101 by broadly describing the claims based on general statements. Hence, the District Court abused its discretion by not considering the specific limitations of Claim 67, particularly the use of a “bounding box”.

G. AWARDING COSTS TO INTEL IS INAPPROPRIATE IN VIEW OF THE ABUSE OF DISCRETION OF THE DISTRICT COURT

The District Court dismissal of the case based on abuse of discretion has created a burden of costs on the Plaintiff-Appellate which is unfair. See DKT 86, 87.

VI. CONCLUSION

For the reasons stated, it is respectfully submitted that the District Court erred in finding Claim 67 unpatentable under Section 101. This error is compounded by the award of costs to Intel.

It is respectfully that this Court find Claim 67 patentable under Section 101, and that the award of costs to Intel be reversed.

Date: August 20, 2014

Respectfully submitted

/s/David Fink
David Fink

Fink & Johnson
7519 Apache Plume
Houston, TX 77071
Telephone: 713 729-4991
Fax: 713 729-8408
texascowboy6@gmail.com

Attorney for Plaintiff-Appellant
FuzzySharp Technologies Inc.

VII. CERTIFICATE OF SERVICE

I hereby certify that on August 20, 2014, I electronically file with the Clerk fo the Court the Opening Brief and Addendum of Plaintiff-Appellant FuzzySharp Technologies Inc. using the CM/ECF, which I understand automatically sends email notification of such a filling to all counsel of record.

/s/ David Fink

David Fink
Attorney for Plaintiff-Appellant

**CERTIFICATE OF COMPLIANCE
UNDER FED. R. OF APP. P. 32(a)(7) AND FED. CIR. R. 32**

Counsel for Plaintiff-Appellant FuzzySharp Technologies Inc. certifies that the Brief contained herein has a proportional spaced 14-point typeface and contains 4845 words, based on the word count feature of WordPerfect, including footnotes and endnotes.

Pursuant to Fed. R. of App. P. 32(a)(7)(B)(iii) and Fed. Cir. R. 32(b), this word count does not include the words contained in the Certificate of Interest, Table of Contents, Table of Authorities, and Statement of Related Cases.

Dated: August 20, 2014

Respectfully submitted,

/s/s David Fink
David Fink
Attorney for Plaintiff-Appellant

ADDENDUM

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA

FUZZYSHARP TECHNOLOGIES INC.,

Plaintiff,

vs.

INTEL CORPORATION,

Defendant.

Case No.: 12-CV-04413 YGR

**ORDER CONSTRUING CLAIM TERMS IN
DISPUTE AND GRANTING SUMMARY
JUDGMENT IN FAVOR OF DEFENDANT INTEL
CORPORATION**

Plaintiff Fuzzysharp Technologies, Inc. (“Fuzzysharp”) brings this patent infringement action against Defendant Intel Corporation (“Intel”) alleging that Intel HD Graphics Code “Sandy Bridge” infringes on U.S. Patent Nos. 6,172,679 (“the ‘679 Patent”) and 6,618,047 (“the ‘047 Patent”) owned by Fuzzysharp. Presently before the Court are two matters for decision: (1) the parties’ claim construction dispute; and (2) the motion of Intel (Dkt. No. 28), originally filed as a motion for judgment on the pleadings, on the grounds that the asserted claims are not patent-eligible under 35 U.S.C. § 101 because the asserted claims of U.S. Patent No. 6,618,047 (“the ‘047 Patent”) (claims 1, 8, 11, 12, 13, 46, 57, 64, 65, and 67) and of U.S. Patent No. 6,172,679 (“the ‘679 Patent”) (claims 1, 4, and 5) (the “Asserted Claims”)¹ are unpatentably abstract. The Court converted the motion to one for summary judgment, and permitted the parties to file supplemental briefing and evidence in support of and in opposition to the motion. (*See* Order Converting Motion for Judgment on the Pleadings Into Motion for Summary Judgment, Dkt. No. 45.)

Having carefully considered the papers submitted and the pleadings in this action, and for the reasons set forth below, the Court **ORDERS** as follows:

¹ At the time of the filing of Intel’s Motion, there were 40 asserted claims. Fuzzysharp subsequently withdrew all but these thirteen. (Dkt. No. 66-2, Exh. B [Declaration of Victoria Q. Smith in, 8/9/2013 email from Dave Fink to Victoria Smith].)

(1) The remaining terms² for construction have the following meaning:

<u>Term</u>	<u>Construction</u>
<p>Group 3('047 claim 1)</p> <p>“determining which of said at least one of 3-D surfaces or their sub-elements is always invisible or always visible”</p>	<p>“determining whether the 3-D surfaces are totally hidden or totally visible”</p>
<p>Group 4 ('679 claim 1)</p> <p>“determining for said viewpoint, whether each said selected surface is</p> <p>(a) an always unoccluded surface, an always hidden surface, or a remaining surface; or</p> <p>(b) an always unoccluded surface, or a remaining surface; or</p> <p>(c) an always hidden surface, or a remaining surface;”</p>	<p>“for each surface and from a predetermined viewpoint:</p> <p>(a) determining whether the surface is a totally visible surface, a totally hidden surface, or a surface that is neither totally visible or totally hidden; or</p> <p>(b) determining whether the surface is a totally visible surface or a surface that is neither totally visible or totally hidden; or</p> <p>(c) determining whether the surface is a totally hidden surface or a surface that is neither totally visible or totally hidden.”</p>

(2) The Asserted Claims are unpatentably abstract as a matter of law and Intel’s Motion for Summary Judgment is **GRANTED**.

BACKGROUND

Fuzzyssharp accuses Intel of infringing U.S. Patent Nos. 6,172,679 (“the ‘679 Patent”) and 6,618,047 (“the ‘047 Patent”). The patents claim methods to reduce the number of visibility computations required to render a three-dimensional scene as a two-dimensional digital computer graphic image. The patents describe methods of analyzing a scene before employing prior art

² Over the course of the claim construction briefing, the parties agreed on constructions for all terms at issue except Group 3 and Group 4.

rendering techniques. In essence, the patent claims describe a method of analyzing which parts of a scene will be visible and which will be hidden, and which may or may not be hidden/visible depending upon the vantage point of the viewer.

I. THE TECHNOLOGY

In 3D computer graphics, computers generate two-dimensional images to represent three-dimensional scenes. In general, a three-dimensional scene is first established by specifying the objects in the scene, how they are lit, and from which point(s) and which angle(s) they will be viewed (“viewpoints”). Producing two-dimensional images of that three-dimensional scene requires the specification of a viewpoint and viewing direction relative to the scene. For a given viewpoint and in scenes with opaque objects only, at most one opaque object or surface can be visible along any single line of sight or viewpoint. For example, in a scene where a cube sits behind a sphere from the perspective of the simulated viewer, the sphere would block the viewer from “seeing” some or all of the cube. (See Figure 1.)

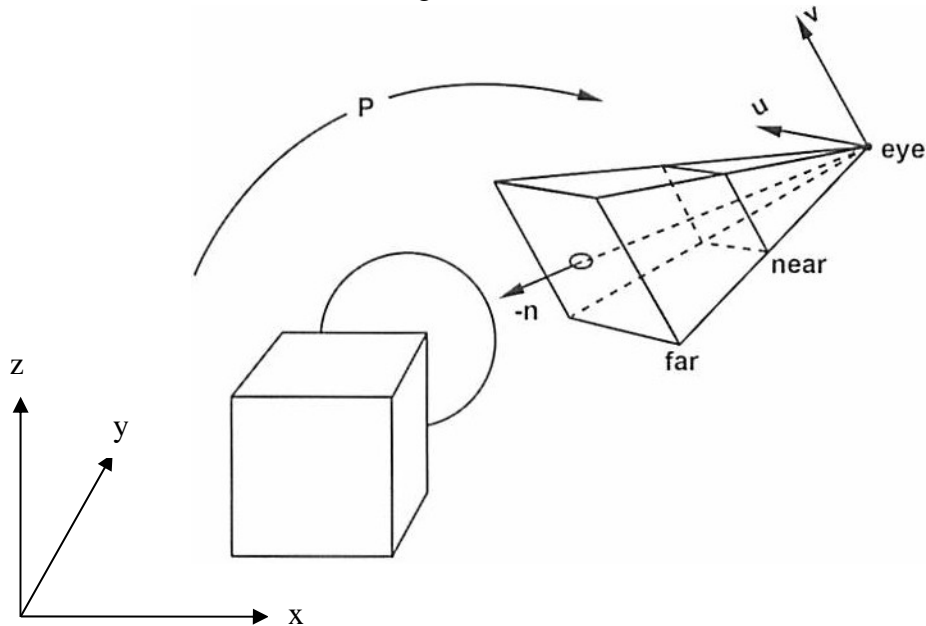
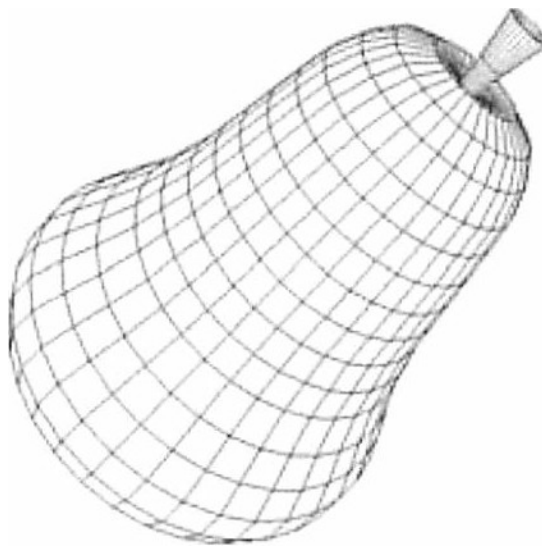


Figure 1: A simulated viewpoint (upper right) viewing a geometric scene model (lower left)

Generally the next steps in rendering a three-dimensional scene as a two-dimensional image are that the properties of the display device are specified, and the coordinates for display of the image on the screen of the device, *i.e.* the “pixels” (points of light with properties that may include color and intensity). The display screen contains a two dimensional array of pixels, each of which

1 displays a color computed for some point in the scene. The pixels collectively form the images
2 displayed on the screen. In order to determine which surface should appear in an image from a
3 given viewpoint, the relative “depth” in the scene (*i.e.*, location on the z-axis) for the various object
4 surfaces must be determined and compared. If objects are opaque and aligned such that one object
5 “blocks” the view of another, then the object that is “nearer” to the viewpoint (*i.e.*, less “depth” in
6 the scene) will have visible surfaces, while the object that is behind the first and “farther” from the
7 viewpoint (*i.e.*, greater “depth” in the scene) will have surfaces hidden by the first.

8 3D graphics are typically rendered using GPUs (graphics processing units). The surfaces
9 that will appear in the image from a particular viewpoint are represented as a collection of basic
10 geometric shapes, or “primitives.” (*See* Figure 2.) A GPU is specialized hardware that receives the



21 **Figure 2: Collection of Primitives For Displaying Visible Surfaces of an Object From a**
22 **Particular Viewpoint**

23 primitives or surfaces from an application program and then displays the primitives or surfaces as
24 an image on the computer screen. Upon receiving a primitive, the GPU converts it into a group of
25 pixels that represents the points on a display device.

26 GPUs typically use frame buffers, which are a portion of computer memory (buffer)
27 organized into a discrete grid (frame) of pixels that store information about each pixel. In the frame
28 buffer, each pixel has memory bits dedicated to representing the color and light intensity to be

1 displayed at the location corresponding to that pixel. The frame buffer is connected to specialized
2 computer hardware that can rapidly scan out the contents of memory to determine what should be
3 displayed.

4 Many computer graphics rendering systems also include a specialized portion of computer
5 memory designated as a “z-buffer” or “depth buffer.” The GPUs can use the z-buffer to store data
6 about the “depth” for each pixel for the purpose of eliminating hidden surfaces at the level of
7 individual pixels or for groups of pixels. The process of using the z-buffer to determine which
8 scene points are visible is known as “z-buffering” or “z-testing.” For scene points that are
9 determined to be visible, the values for color and other properties will be written into the frame
10 buffer and become the values for the corresponding pixel. For a scene point that is determined to
11 be hidden, the values for the color and other properties of the point will not be written into the
12 frame buffer, and the data for the pixel that corresponds to that point will not be overwritten. Thus,
13 the frame buffer and z-buffer are used together to determine what should be displayed on the
14 screen.

15 **II. THE PATENTS**

16 The patents-in-suit involve 3D computer graphics and purport to disclose “an improved
17 method for performing visibility calculations.” (‘679 Patent Title, Field of Invention, col. 2:17- 20.)
18 The patents describe the background art, and the problem to be solved by the patents, in part as
19 follows:

20 Visible surface detection is one of the most basic operations in 3D graphics. It
21 is applied to generate images of surfaces directly visible to a viewer. . . . The
22 standard strategy of visible surface detection is to divide surfaces into patch
23 elements and compare the spatial relationship between these elements. Using
24 this strategy, the visibility of surfaces cannot be determined until they have been
25 analysed in detail. Although many techniques have been developed to address
26 this issue, none are ideal as they either still require elaborate analysis of
27 surfaces, or they impose various restrictions to the scene. The limitations of the
28 current techniques can seriously affect the speed of visible surface computation.
If the scene is complicated, many surfaces may be invisible. However, the
image generation is often slowed down by the need to analyse each surface
element in detail.

(‘047 Patent, col. 1:18-36.)

1 The '047 Patent is a continuation of the '679 Patent and shares a nearly identical
2 specification. The only difference between the patents lies in the Summary of the Invention, which
3 was modified in the '047 Patent to reflect differences in the claims in the patents. The Asserted
4 Claims (claims 1, 4, and 5 of the '679 Patent, and claims 1, 8, 11, 12, 13, 46, 57, 64, 65, and 67 of
5 the '047 Patent) describe a method (and variations) for reducing the number of visibility
6 computations a GPU is required to perform in order to render a three-dimensional computer graphic
7 image. The method reduces computations by pre-processing the surfaces from a group of
8 viewpoints by: predetermining the visibility of surfaces from a group of viewpoints; creating data
9 structures storing the resulting visibility computations; and then ignoring the pre-identified surfaces
10 during later visibility computations, such as the visibility computations performed during rendering.

11 Claim 1 of the '047 Patent is illustrative:

- 12 1. A method of reducing the visibility related computations in 3-D computer
13 graphics, the visibility related computations being performed on 3-D surfaces
14 or their sub-elements, or a selected set of both, the method comprising:
15 identifying grid cells which are under or related to the projections or extents of
16 projections associated with at least one of said 3-D surfaces or their sub-
17 elements;
18 comparing data associated with said at least one of 3-D surfaces or their sub-
19 elements with stored data associated with the grid cells;
20 determining which of said at least one of 3-D surfaces or their sub-elements is
21 always invisible or always visible to a viewpoint or a group of viewpoints
22 by projection based computations prior to a visibility computation; and
23 ignoring said determined at least one of the 3-D surfaces or their sub-elements
24 during said visibility computation.

25 ('047 Patent, col. 27:66-28:17.) The claim language refers to pre-processing the visibility of
26 surfaces (or parts of surfaces) before traditional rendering ("prior to a visibility computation"). The
27 claimed method creates a set of fixed, imaginary planes onto which the surfaces or parts of surfaces
28 of the objects in the scene are projected ("projections"), imposes on each plane an imaginary grid
consisting of abstract individual cells ("grid cells"), creates a data structure that stores depth data
for each abstraction ("stored data associated with the grid cells"), and then compares the data for a
given surface with the data for the corresponding grid cell to determine whether that surface is
totally invisible or totally visible. After such pre-processing, an application program can identify

which surfaces are totally invisible or totally visible from a given viewpoint group by consulting the data structures associated with the viewpoint group. The application program can then “ignore” (‘047 Patent claim 1), “skip” (‘047 Patent claims 11 and 12), or “exempt” (‘679 Patent claim 1) totally invisible surfaces when sending data to the GPU for rendering an image. Alternatively, the program could send data for totally visible (“always unoccluded”) surfaces to the GPU for rendering but could “ignore,” “exempt,” “skip,” or otherwise direct the GPU not to perform the z-buffering tests for such surfaces. (‘679 Patent claim 1 and ‘047 Patent claims 1, 46.)

The written description likewise describes pre-processing surfaces and for each viewpoint group: classifying the surfaces into categories, creating data structures storing that classification, and then using the stored data to exempt totally visible and totally invisible surfaces from further visibility computations. (*See* ‘679 Patent col. 14:13-25 [“In subsequent rounds of computations, where the form factors between patches have to be re-evaluated, the fuzzy projection computations are not repeated. The totally visible/non-hiding and the totally invisible patches of each-viewpoint can be simply retrieved from the stored information”].)

The overarching goal of the patents is to “reduc[e] the complexity of visibility calculations required for the production of multi-dimensional computer generated images.” (‘679 Patent col. 28:25-27.) The parties have agreed that this means “decreasing the number of computational operations to perform visibility computations.” (Dkt. No. 64 at 5.) By analyzing a scene from a viewpoint or group of viewpoints at the outset and creating data structures reflecting the visibility of surfaces or portions of surfaces rather than individual pixels or groups of pixels, the claimed method was designed to reduce the portions of a scene that had to be subjected to the full-blown visibility and rendering computations that were known and used in the prior art.

CLAIM CONSTRUCTION

For purposes of claim construction, the sole dispute left between the parties is whether the claim terms require a 3-part categorization (*i.e.*, totally hidden, totally visible, and remaining) or whether they are satisfied by a 2-part determination only (*i.e.*, totally hidden vs. remaining, totally visible vs. remaining, or totally visible vs. totally hidden).

Both Group 3 and Group 4 claims concern categorization of surfaces for pre-processing. The parties dispute whether the claims require a three-part determination or a two-part determination. The parties competing constructions follow, with the differences in bold and italics:

<u>Term</u>	<u>Fuzzysharp's Proposed Construction</u>	<u>Intel's Proposed Construction</u>
Group 3 “determining which of said at least one of 3-D surfaces or their sub-elements is always invisible or always visible” (‘047 claim 1)	“determining which of the 3-D surfaces are <i>either</i> hidden or visible <i>from a single viewpoint, or a group of viewpoints.</i> ”	“determining whether the 3-D surfaces are totally ³ hidden or totally visible”
Group 4 “determining for said viewpoint, whether each said selected surface is (a) an always unoccluded surface, an always hidden surface, or a remaining surface; or (b) an always unoccluded surface, or a remaining surface; or (c) an always hidden surface, or a remaining surface;” (‘679 claim 1)	“for each surface and from a predetermined viewpoint, <i>carrying out one of the following groups of tests in (a), or (b), or (c) to determine, respectively:</i> (a) <i>which single one of the following categories the surface falls:</i> a totally visible surface, a totally hidden surface, or a surface that is neither totally visible or totally hidden; (b) <i>which single one of the following categories the surface falls:</i> a totally visible surface or a surface that is neither totally visible or totally hidden; or (c) <i>which single one of the following categories the surface falls:</i> a totally hidden surface or a surface that is neither totally visible or totally hidden.”	“for each surface and from a predetermined viewpoint: (a) <i>determining whether the surface is</i> a totally visible surface, a totally hidden surface, or a surface that is neither totally visible or totally hidden; <i>or</i> (b) <i>determining whether the surface is</i> a totally visible surface or a surface that is neither totally visible or totally hidden; <i>or</i> (c) <i>determining whether the surface is</i> a totally hidden surface or a surface that is neither totally visible or totally hidden.”

³ Although Fuzzysharp’s proposed construction for claim 1 of the ‘047 patent does not indicate that the terms “hidden” and “visible” mean “totally hidden” and “totally visible,” the parties have agreed that they do. (Dkt. No. 64 at 1-2.)

Thus, when Claim 1 of the '679 Patent recites:

...for selected ones of said surfaces, determining for said viewpoint whether each said selected surface is

(a) an always unoccluded surface, an always hidden surface, or a remaining surface; *or*

(b) an always unoccluded surface, or a remaining surface; *or*

(c) an always hidden surface, or a remaining surface; ...

('679 Patent, col. 28:33-39), the parties agree that the italicized "or's" are disjunctive, such that only one of the three sets of tests ((a), (b), and (c)) need be performed. Their dispute lies within each of (a), (b), and (c).

Under Intel's construction, the method of the patents pre-processes the scene to determine from a viewpoint or group of viewpoints which surfaces of the scene are totally hidden, which are totally visible, and which are "remaining." Intel thus proposes that "remaining" means "neither totally hidden nor totally visible" wherever it is used in the patents. Using those data structures, a program may then avoid sending the totally hidden surfaces to the graphics processing unit ("GPU") for rendering, since a totally hidden surface need not be rendered at all. Similarly, the program can send the totally visible surfaces to the GPU, but instruct that no further visibility computations need to be performed to render the surface. Thus, the method of the patents tests for two categories of surfaces, totally hidden or totally visible, and any surface not meeting either of those categories falls into a third, "remaining surfaces." The remaining surfaces are those surfaces that will need additional visibility computations performed by the GPU prior to or during rendering. According to the patents, this categorization of surfaces reduces the number of visibility computations that the GPU must perform, leading to an overall gain in system efficiency.

According to Fuzzysharp, Intel's interpretation of the term "remaining" is inaccurate and leads to confusion. The patents describe different possible operations that test whether surfaces are totally visible or totally invisible. The simpler operations described test only for one type of surface. For example, '679 Patent, Claim 1(c) tests only for hidden surfaces, and under that test a surface is either totally hidden or not. Under this test, Fuzzysharp contends, the "not" is the "remaining surface." Likewise, the operation described in the '679 Patent, Claim 1(b) is to test for totally visible (or unoccluded) surfaces. The nature of this test is that, if the surface fails the test for

being visible, then it is classified as being a “remaining” surface, in this case one that can be described as either: (1) totally hidden; or (2) neither totally hidden nor totally visible. Under ‘679 Patent Claim 1(a) tests are carried out for both totally visible *and* totally hidden surfaces, and failing both tests also means that the surface is “remaining.” However, in this scenario “remaining surfaces” would *only* be neither totally hidden nor totally visible. Fuzzysharp further contends that the ‘047 Patent, Claim 1 describes only two simple operations: a test for always hidden (invisible) surfaces *or* always visible surfaces; those surfaces failing the test used are classified as remaining surfaces. In Fuzzysharp’s reading of the claim language, a “remaining” surface is a simply a by-product of steps for determining the information necessary to simplify computations, and therefore means different things depending upon the test used.

I. CLAIM LANGUAGE

Unlike Fuzzysharp’s proposed construction, Intel’s construction gives meaning to the limitations in the claims themselves. First, looking at the disputed portion of the claim in context supports giving a single, consistent meaning to the term “remaining surface” as a category distinct from both “hidden” and “visible/unoccluded.” The language of Claim 1 recites, in part:

...determining for said viewpoint whether each said selected surface is
 (a) an always unoccluded surface, an always hidden surface, or a remaining surface; or
 (b) an always unoccluded surface, or a remaining surface; or
 (c) an always hidden surface, or a remaining surface;
wherein said remaining surface is a surface which is unable to be determined with certainty as to whether it is either unoccluded or hidden

(‘679 Patent, col. 28:33-42, emphasis added.) Thus, the “wherein” clause is written to apply to *each* of (a), (b), and (c), and to provide a definition of a “remaining surface” for all, not just a definition that applies only to (a). When a “wherein” clause “expresses the inventive discovery,” rather than just stating a result of certain limitations, it is definitional. *See Griffin v. Bertina*, 285 F.3d 1029, 1034 (Fed.Cir. 2002); *Intergraph Hardware Technologies Co. v. Toshiba Corp.*, 508 F. Supp. 2d 752, 769 (N.D. Cal. 2007) (“wherein” clause stated limitations that were part of construction rather than expressing result of limitation).

1 The other limitations of Claim 1 are consistent with Intel's reading. First, Claim 1 requires
2 "exempting from said occlusion or invisibility relationship computation those surfaces which are
3 *either* always unoccluded *or* always hidden." ('679 Patent col. 28:43-45, emphasis added). The
4 "exempting" limitation can be met only if a surface has been tested sufficiently to determine
5 whether it is always visible or always hidden, rather than testing for only one of these categories, as
6 Fuzzysharp suggests. Second, Claim 1 defines "*said remaining surfaces*" in the "wherein" clause
7 and then goes on to require "maintaining a record of *said remaining surfaces*." ('679 Patent col.
8 28:40-42, 46.) The "maintaining" limitation cannot be met unless surfaces have been tested
9 sufficiently to determine those that are *neither* always visible nor always hidden and thus fall into
10 the third category, "remaining," as to which a record is to be maintained.

11 Fuzzysharp's proposed construction makes '679 Patent Claim 1 clauses (b) and (c)
12 redundant of clause (a). Fuzzysharp proposes that '679 Patent Claim 1(a) be construed to mean
13 carrying out a test to determine into which *single* one of the following categories the surface falls: a
14 totally visible surface, *or* a totally hidden surface, *or* a surface that is neither totally visible or
15 totally hidden. In other words, a test under Claim 1(a) tests for only one condition, with a binary
16 result (*i.e.*, meets category or does not meet category). Under that construction, however, clauses
17 (b) and (c) would have no meaning, because they would not cover a case not already expressed in
18 clause (a).

19 Intel's construction gives effect to all three clauses since clause (a) tests for all three
20 categories (totally visible, totally hidden, and "remaining"), clause (b) tests for two of the three
21 categories (totally visible and "remaining," with all others being totally hidden), and clause (c) tests
22 for a different set of two of the three (totally invisible and "remaining," with all others being wholly
23 visible). Thus, Intel's construction fits with the language of '679 Patent Claim 1 itself while
24 Fuzzysharp's does not.

25 Though the '047 Patent claims do not use the term "remaining surfaces," the same
26 categorization is implicit because those claims simply require determining the totally hidden and
27 totally visible surfaces and then exempting *both* those categories of surfaces from future visibility
28 computations by the GPU. In both cases, the GPU must then perform visibility computations only

on the “remaining surfaces.” The clause in dispute requires “determining which of said at least one of 3-D surfaces or their sub elements is always invisible or always visible.” But the final step requires “ignoring said determined at least one of the 3-D surfaces or their sub-elements during said visibility computations.” (‘047 Patent, col. 28:9-15.) The claim thus requires testing to determine which surfaces are “always invisible or always visible” so that both wholly hidden and wholly visible surfaces are excluded. Read in context, the “or” does not suggest that only one test need be performed; it instead signifies that a surface cannot be both “always invisible” and “always visible.”

In sum, in order for the claim language itself to be internally consistent, the language in both the ‘679 Patent, Claim 1, and the ‘047 Patent, Claim 1 require a construction that means the method can test in such a way as to result in three categories of surfaces, not just two.

II. PATENT SPECIFICATION

Fuzzyssharp’s position is, essentially, that reading the patent to require determining both “always visible” and “always invisible” for all surfaces would not result in the reduction of visibility-related computations that is part of the goal of the patent, as expressed in the Summary of Invention for both patents. (‘679 Patent, col. 2: 22-65, ‘047 Patent, col. 2:22-46.) However, Intel correctly notes that the patent specification in the ‘679 Patent itself states:

In the first round of computations, all the fuzzy projection and normal hemicube computations for every receiving patch are carried out as required. ***During these computations, the emission patches at each group/viewpoint are classified into three groups: totally visible/non-hiding, totally invisible, and the remainder.*** This classification information is stored using data structures such as arrays or linked lists.

(‘679 Patent, col. 14:13-19, emphasis added; *see also* ‘047 Patent ,col. 13:46-50 [same]).

Elsewhere, the specification similarly describes testing for both total invisibility and total visibility, and all others into a “remaining” category:

“[a]fter the filtering off of totally invisible patches by the invisibility technique, the visibility technique detects totally visible and non-hiding patches. ... The remaining patches are not totally invisible. They are also not totally visible/non-hiding.”

(‘679 Patent, col. 14:59-65; *see also* ‘047 Patent, col. 14:28-29 [same].) Thus the specification supports a construction requiring that the claims will, at least in some cases, put surfaces into three categories. Contrary to Fuzzysharp’s argument, such a construction is also consistent with the goal of eliminating a full visibility computation for surfaces already known to be wholly visible or wholly hidden. Thus the specification supports Intel’s construction reading the claim language in both the ‘679 Patent, Claim 1, and the ‘047 Patent, Claim 1 to result in a determination of three categories of surfaces.

III. PROSECUTION HISTORY

While not necessary to the Court’s construction determination, Intel’s proposed construction is consistent with the prosecution history stated in the patents. The patent examiner initially rejected the claim language in the ‘047 Patent which stated “determining which of the 3-D surfaces or their sub-elements are always invisible or always visible.” (Dkt. No. 66-2 to 66-6 [Declaration of Victoria Q. Smith], Exh. C [‘047 Patent File History, Application] at 48.) The patent examiner found that the claim was obvious in light of the prior art in *Kadota*, which disclosed a determination whether 3-D surfaces were always visible or invisible from a viewpoint. The rejection stated:

As per independent claim 1, Kadota discloses determining 3D surfaces that are always visible or invisible to a viewpoint by projection (abstract) and treating those surfaces differently from remaining surfaces ..., it would have been obvious to one of ordinary skill in the art at the time of invention to use the disclosure of Kadota because he teaches transmitting and processing predetermined projection coordinates to a hidden surface processor *which evaluates both visible and invisible surfaces*, transmits the visible surfaces and ignores the invisible surfaces.

(Smith Dec., Exh. E [‘047 Patent File History, September 22, 2000 Rejection] at 2, emphasis added.)

Previously, the Patent and Trademark Office had allowed the ‘679 Patent, the Reasons for Allowance having stated:

the applicant claims sorting surfaces as visible, hidden or remaining (either visible or nonvisible) and processing the remaining surfaces to perform and

reduce visible/hidden surface computations, which is not disclosed in the prior art of record.

(Smith Dec., Exh. D [‘679 Patent File History, November 15, 1999 Notice of Allowance] at 2.) Fuzzyssharp does not address this prosecution history or offer any contradictory arguments.

Thus, based on the prosecution history offered by Intel, the Patent and Trademark Office appears to have considered the three categories of surfaces to be critical in distinguishing the patents over prior art.

THE COURT’S CONSTRUCTION:

Based upon the foregoing, the Court interprets the claims at issue as follows:

<u>Term</u>	<u>Construction</u>
Group 3(‘047 claim 1) “determining which of said at least one of 3-D surfaces or their sub-elements is always invisible or always visible”	“determining whether the 3-D surfaces are totally hidden or totally visible”
Group 4 (‘679 claim 1) “determining for said viewpoint, whether each said selected surface is (a) an always unoccluded surface, an always hidden surface, or a remaining surface; or (b) an always unoccluded surface, or a remaining surface; or (c) an always hidden surface, or a remaining surface;”	“for each surface and from a predetermined viewpoint: (a) determining whether the surface is a totally visible surface, a totally hidden surface, or a surface that is neither totally visible or totally hidden; or (b) determining whether the surface is a totally visible surface or a surface that is neither totally visible or totally hidden; or (c) determining whether the surface is a totally hidden surface or a surface that is neither totally visible or totally hidden.”

//

//

SUMMARY JUDGMENT

Intel brings its motion for summary judgment on the grounds that, pursuant to 35 U.S.C. § 101, the Asserted Claims are not patent-eligible. Intel argues that the Asserted Claims set forth an abstract idea without adding more than an instruction to “apply it,” or recite an abstract idea without requiring any application containing an “inventive concept,” or both. Whether the claims satisfy 35 U.S.C. § 101’s criteria for patent-eligible subject matter is a question of law for the Court. *Dealertrack, Inc. v. Huber*, 674 F.3d 1315, 1333 (Fed. Cir. 2012). The Federal Circuit has held that “any attack on an issued patent based on a challenge to the eligibility of the subject matter must be proven by clear and convincing evidence.” *Ultramercial, Inc. v. Hulu, LLC*, 722 F.3d 1335, 1342 (Fed. Cir. 2013).

Here, there are no disputed issues of fact raised by the parties.⁴ The parties agreed on constructions for all of the claim terms in dispute but two. Those two terms were the subject of claim construction briefing and argument, and their constructions are set forth above, in the first part of this Order.

I. LEGAL FRAMEWORK

In analyzing patentability under Section 101, a court must first identify whether the claimed invention fits within one of the four statutory classes set forth in the statute. Section 101 specifies four broad, independent categories of inventions or discoveries that are patent eligible: “process, machine, manufacture or composition of matter.” 35 U.S.C. § 101. Second, a court must assess whether the claimed invention falls into any of the three specific exceptions established by Supreme Court precedent: “laws of nature, physical phenomena, and abstract ideas.” *Diamond v. Chakrabarty*, 447 U.S. 303, 309 (1980). These exceptions have grown out of the Supreme Court’s

⁴ Intel initially brought this motion as one for judgment on the pleadings. (Dkt. No. 28.) Based upon the arguments made in opposition by Fuzzysharp, the Court converted the motion to one for summary judgment and permitted the parties to file supplemental briefing and evidence, and specifically directed that the parties identify any terms or phrases that required a claim construction prior to a decision on summary judgment. (Dkt. No. 45.)

“concern that patent law not inhibit further discovery by improperly tying up the future use of laws of nature,” natural phenomena, and abstract ideas. *Mayo Collaborative Servs. v. Prometheus Labs., Inc.*, 132 S. Ct. 1289, 1301 (2012). “A principle, in the abstract, is a fundamental truth...[which] cannot be patented.” *Gottschalk v. Benson*, 409 U.S. 63, 67 (1972) (internal citations and quotations omitted) (“*Benson*”). “Phenomena of nature, though just discovered, mental processes, and abstract intellectual concepts are not patentable, as they are the basic tools of scientific and technological work.” *Id.* Included within the exception for abstract ideas are such concepts as mathematical formulae. *Id.* at 68.

A court must then determine whether the patent claims “pose any risk of preempting an abstract idea.” *CLS Bank Int’l v. Alice Corp. Pty. Ltd.*, 717 F.3d 1269, 1282 (Fed. Cir. 2013). Even if the patent claims rely on an abstract concept such as a mathematical formula, the “balance of the claim can be evaluated to determine whether it contains additional substantive limitations that narrow, confine, or otherwise tie down the claim so that, in practical terms, it does not cover the full abstract idea itself.” *Id.* at 1282 (citing *Mayo*, 132 S.Ct. at 1300; *Bilski v. Kappos*, 130 S. Ct. 3218, 3231 (2010); and *Diehr*, 450 U.S. at 187).

A. Supreme Court Authorities

Over time, the Supreme Court has set forth a number of different factors to consider in distinguishing a patentable invention from an unpatentable abstract concept. Six cases are particularly instructive here. First, in *Benson*, the Supreme Court held that a mathematical formula for converting binary-coded decimal numerals into pure binary code *was not* patentable because the process claimed was “so abstract and sweeping” that it could cover usage in a variety of operational settings and could be performed through any existing or future machinery, or without machinery at all. *Benson*, 409 U.S. at 67. The fact that the process or method is carried out with the use of common computer components, such as computer memory, does not render an otherwise abstract concept patentable and processes that “can be carried out in existing computers long in use, no new machinery being necessary” are ineligible for patent protection. *Id.* at 67.

Following thereafter, in *Parker v. Flook*, 437 U.S. 584 (1978) (“*Flook*”), the Supreme Court held that a patent which described a mathematical formula for calculating alarm limit values *was*

1 *not* patentable due to the abstract nature of the subject matter. There, the patentee sought to
2 distinguish *Benson* by arguing that, although there were a number of “post-solution” uses for the
3 formula, the patent included specific “post-solution” activity to make the formula useful in the
4 petrochemical and oil-refining industries. The Supreme Court rejected this distinction, stating that

5 [t]he notion that post-solution activity, no matter how conventional or obvious
6 in itself, can transform an unpatentable principle into a patentable process exalts
7 form over substance. A competent draftsman could attach some form of post-
8 solution activity to almost any mathematical formula; the Pythagorean theorem
9 would not have been patentable, or partially patentable, because a patent
application contained a final step indicating that the formula, when solved,
could be usefully applied to existing surveying techniques.

10 *Flook*, 437 U.S. at 590 (1978). Thus, while a claimed invention does not lose patent eligibility
11 simply because it relies on a phenomenon of nature or mathematical formula, such a phenomenon
12 or formula cannot, on its own, be patentable without “some other inventive concept in its
13 application.” *Id.* at 594.

14 Subsequently, by contrast, the Supreme Court found patentable under Section 101 a process
15 combining an abstract formula with practical applications and limitations in *Diamond v. Diehr*, 450
16 U.S. 175 (1981) (“*Diehr*”). There, the patent claimed a process for curing synthetic rubber that,
17 while involving an abstract mathematical formula, used that formula in combination with a novel
18 set of steps to be followed in a particular application. The overall process was patentable because it
19 integrated the equation into the rubber curing process as a whole, using constant temperature
20 readings inside the mold, and making repeated recalculations to determine the optimum time to
21 open the rubber press. *Id.* The Court held that patent avoided the abstractness exception to patent
22 eligibility since the patentees did not “seek to pre-empt the use of that equation. Rather, they seek
23 only to foreclose from others the use of that equation *in conjunction with all of the other steps* in
24 their claimed process.” *Id.* at 187 (emphasis added). Thus an abstract idea, law of nature, or
25 mathematical formula could not be patented but “an *application* of a law of nature or mathematical
26 formula to a known structure or process may well be deserving of patent protection.” *Id.*

27 In its 2010 *Bilski* decision, the Supreme Court reiterated these principles. *Bilski*, 130 S. Ct.
28 at 3231. The Court held that the Federal Circuit’s “machine-or-transformation test,” while not

dispositive of patent eligibility, provides a “useful tool” for determining whether a claimed method or process is patent eligible. *Bilski*, 130 S. Ct. at 3226-27. Under that formulation, a method or process is patent eligible when: “(1) it is tied to a particular machine or apparatus, or (2) it transforms a particular article into a different state or thing.” *Id.* at 3225 (quoting *In re Bilski*, 545 F.3d 943, 954 (Fed. Cir. 2008) (*en banc*)). In *Bilski*, the Court found that claims describing the concept of hedging and a mathematical formula for hedging against risk in price fluctuations for buyers and sellers of commodities were not patentable. The Court held that “hedging” was a basic economic concept, and reducing the concept to a mathematical formula did not take it out of the realm of abstract ideas. *Id.* Likewise, adding claims that simply applied the formula to particular markets was not enough to convert the abstract concept into a patentable one, just as adding a field of use or common post-solution components was not sufficient in *Flook*. *Id.*

Most recently, the Supreme Court held that claims were too abstract, and not patentable under section 101, where “the claimed processes (apart from the natural laws themselves) involve well-understood, routine, conventional activity previously engaged in by researchers in the field....” *Mayo*, 132 S. Ct. at 1294. The Court held that a claim must apply an abstract idea in a way that reflects an “inventive concept,” which it illustrated by distinguishing its decisions in *Flook* and *Diehr*. In *Flook*, the claimed method used a computer to calculate alarm-limit values to signal dangers in operating a catalytic converter, but the process did “nothing other than ‘provid[e] an unpatentable formula for computing [the] updated alarm limit.’” *Mayo*, 132 S. Ct. at 1299 (quoting *Flook*, 437 U.S. at 586) (internal quotations and brackets omitted). All the other steps—the chemical processes involved in catalytic conversion, the monitoring of chemical process variables, the use of alarm limits to signal danger, and the adjusting of those limits through the use of computers for “automatic monitoring-alarming”—were “‘well known.’” *Id.* at 1299 (quoting *Flook*, 437 U.S. at 594). Thus, “putting the [abstract] formula to the side, there was no ‘inventive concept’ in the claimed application of the formula.” *Id.* In *Diehr*, by contrast, the “other steps” added something “that transformed the process into an inventive application of the formula.” *Mayo*, 132 S Ct. at 1299 (process was patentable because the other steps, and the combination of them, were not “obvious, already in use, or purely conventional”).

B. Recent Federal Circuit Decisions

The Federal Circuit has applied these principles in a recent cases involving computing technologies. In one, the Federal Circuit found unpatentable a claimed method which disclosed a number of computer components needed to carry it out. *Accenture Global Servs., GmbH v. Guidewire Software, Inc.*, 728 F.3d 1336, 1343 (Fed. Cir. 2013) (components required included “CPU, ROM, RAM, I/O Adapter, Communication Adapter, Display Adapter, and a User Interface Adapter”). The listed computer components constituted what was described in the patent as a “representative hardware environment.” *Id.* The Federal Circuit found such components insufficient to provide meaningful limitations on the abstract concept of the patent itself. *Id.* “[S]imply implementing an abstract concept on a computer, without meaningful limitations to that concept, does not transform a patent-ineligible claim into a patent-eligible one.” *Id.* at 1345 (citing *Bancorp*, 687 F.3d at 1280). “[T]he complexity of the implementing software or the level of detail in the specification does not transform a claim reciting only an abstract concept into a patent-eligible system or method.” *Id.*

Accenture echoed other recent Federal Circuit decisions finding that simply adding a storage or calculation step performed by a digital computer component did not limit an otherwise abstract concept sufficiently to make it patentable. *See Bancorp Servs., L.L.C. v. Sun Life Assur. Co. of Canada (U.S.)*, 687 F.3d 1266, 1278 (Fed. Cir. 2012) (computer and “high density removable storage means ... such as a compact disc” required by patent claims used only for basic computing functions and repetitive calculations did not impose meaningful limits on the scope of those claims to make them patentable); *Dealertrack, Inc. v. Huber*, 674 F.3d 1315, 1333 (Fed. Cir. 2012) (“simply adding a ‘computer aided’ limitation to a claim covering an abstract concept, without more, is insufficient to render the claim patent eligible”); *Cybersource Corp. v. Retail Decisions, Inc.*, 654 F.3d 1366, 1376 (Fed. Cir. 2011) (added computer storage step naming particular type of storage component does not render otherwise unpatentable process patent-eligible) (citing *Benson*, 409 U.S. at 73-74).

By contrast, the Federal Circuit in *Ultramercial* found claims concerning a method for distributing media products over the Internet to be patent-eligible. *Ultramercial*, 722 F.3d at 1352-

53. There, the claims required steps to be performed through an “extensive computer interface” involving “eleven separate and specific steps with many limitations and sub-steps in each category.” *Id.* There, the Federal Circuit reiterated that “[m]ere reference to a general purpose computer will not save a method claim from being deemed too abstract to be patent eligible.” *Id.* at 1348 (citing *Bilski*, 130 S.Ct. at 3227). However, it held that the requirements of the patent claims at issue in *Ultramercial* meaningfully limited the abstract formulae and algorithms at the core of the claims so that the claimed invention was not “so manifestly abstract as to override the statutory language of section 101.” *Id.* (quoting *Research Corp. Technologies, Inc. v. Microsoft Corp.*, 627 F.3d 859, 869 (Fed. Cir. 2010) [upholding claimed method for rendering a halftone image by manipulation of computer data and output]).

It is with these principles and authorities in mind that the Court turns to the question of patent eligibility presented here.

II. APPLICATION TO PATENTS-IN-SUIT

The Court agrees that the claims here do not state patentable subject matter. The patents describe their invention as “an improved method for performing visibility calculations” in three-dimensional graphics. (‘047 Patent, col. 2:23-24; ‘679 Patent (‘679 Patent), col. 2:19-20; patent Titles.) Visibility calculations are used for “visible surface detection,” which is “one of the most basic operations” in 3D graphics. (‘047 Patent, col. 1:18-21; ‘679 Patent, col. 1:16-18.) The claimed methods of the patents shorten the calculations by *identifying* the parts of the surfaces in a 3-D representation that are always visible or always invisible from a set of viewpoints, and then *skipping* the calculations for that part in further rendering of the image. The ‘047 and ‘679 preambles claim methods of reducing “visibility calculations” (or “computations”) “in 3-D computer graphics” (‘047) and “required for the production of multidimensional computer generated images” (‘679). As agreed by the parties, these phrases mean: “decreasing the number of computational operations required to perform prior art visibility computations” “in the field of

three-dimensional computer graphics” (‘047) or “to produce a 3-D image.” (‘679). (Dkt.No. 64 at 5.)⁵

All of the Asserted Claims adhere to the same basic format as independent claim 1 of the ‘047 Patent. First, the claims include a preamble that states that the method is for reducing the visibility related computations calculations in a certain field of use (such as “3-D computer graphics”), followed by method steps to perform the reduction. The method steps have three components:

(1) mathematical manipulations of data, such as “identifying,” “comparing,” or “determining;”

(2) a reducing step (“ignoring” or “exempting” or “skipping” data or calculations); and

(3) an indication of where these steps fit into the visibility computations.

Finally, there are a few references to generic computing structures (*e.g.*, “data,” a “record,” or a “computer”). In sum, the patents claim a mathematical principle for eliminating or simplifying portions of otherwise well-known and established computations by pre-determining a portion of the calculations, and then specifying that a portion of the calculations can be skipped or omitted going forward. The claims specify no physical output or programming result. Rather, in every claim, the last method step recites either ignoring part of the visibility computations or completing such calculations without the unnecessary steps.

⁵ The parties have defined “visibility calculations,” in relevant part, as “rendering computations that determine which pixels ... of 3D surface(s) are visible,” and thus such calculations already imply the production of a 3D image. (Dkt. No. 64 at 2.) The parties also agreed on the following definitions of terms:

“projections” and “projection planes” refer to “representations of a 3D object’s surfaces on an imaginary plane”;

“grid cells” are an “imaginary lattice structure”;

a “bounding volume” is “the smallest imaginary right quadrangular prism just enclosing the 3-D object’s surfaces”; and

“surfaces” are simply “surfaces” of 3D objects that are part of a scene.

(Dkt. No. 64 at 1-3.)

1 **A. Lack of Meaningful Limitation To A Specific Application/Inventive Concept**

2 The claims do not limit the mathematical formula to any specific use beyond the broad field
3 of invention of “computer graphics.” (‘047 Patent, col. 1:11-16; ‘679 Patent, col. 1:8-13; ‘679 and
4 ‘047 Patents, Abstract [the disclosed method operates “in 3D graphics systems”].) Simply limiting
5 the use of an abstract formula to a particular field is insufficient to avoid the prohibition on
6 patenting abstract ideas. *Bilski*, 130 S.Ct. at 3230; *Mayo*, 132 S.Ct. at 1301.

7 Similarly, the claims only state a reduction in computations, with no specific output. The
8 claims cover any application for which such computations might be used. Any equations having
9 the same “mathematical relationship of entities” fall within the invention regardless of the physical
10 properties or variables to which they are applied. (‘047 Patent, col.4:6:11; ‘679 Patent, col.4:36-
11 40.) Just as the method of converting binary numbers in *Benson* would cover all computer
12 applications of that algorithm, the claims at issue here would cover all applications of their
13 algorithm in the field of “3D computer graphics.” Indeed, the patent specification indicates that the
14 applications of the formula are numerous and would include any process that requires visible
15 surface detection, such as “radiosity calculations to compute the energy interactions between
16 surfaces,” “Hidden Surface Computations” for “computer animation, flight simulation or dynamic
17 graphics,” “Ray Tracing Computations,” “Computer Vision,” “virtual reality applications,” and the
18 “processing and display of scientific data such as energy spectra data.” (‘047 Patent, col. 1:20-22,
19 15:33-41, 15:58-16:19, 16:21-59, 18:33-46, 18:46-48, 18:56-59.)

20 **B. Conventional Post-Solution Activity**

21 Moreover, the claims do not add any steps other than conventional “post-solution” activity
22 to the abstract formula described, making them unpatentable as stated in *Flook* and *Mayo*. As the
23 patent specifications explain, “[v]isible surface detection is one of the most basic operations in 3D
24 graphics” for generating 3D images. (‘047 Patent, col. 1:18-21, 2:25-41; ‘679 Patent, col. 1:16-18,
25 2:20-27.) The patents purport to make surface detection computations faster by computing the
26 “surface elements obviously visible or invisible to each other,” and ignoring those elements,
27 thereby “reducing the visibility related computations.” (‘047 Patent, col. 1:40-43, 2:21-44; ‘679
28 Patent, col. 1:38-40, 2:18-28.) The claims confirm that conventional visibility computations follow

the method steps. Claim 1 of the ‘679 Patent recites that the entire method is performed “prior to an occlusion or invisibility relationship computation (*known per se*) being carried out.” (‘679 Patent, col. 28:31 (emphasis added).) The parties agree that this refers to “prior art types of rendering computations that determine which pixels or groups of pixels of 3D surface(s) are visible, which include z-buffering and other visibility tests performed during rendering.” (Dkt. No. 64 at 2.) The claims simply recite the steps of determining what part of well-known visibility computations may be omitted, omitting them, and carrying out the remaining computations as before. They do not alter the remainder of the rendering process of which those computations are a part.

Thus, the steps that follow are visibility computations that would be performed even if the claimed method were not used. There is nothing in the patents’ claims or specifications that adds to the formula steps, or a combination of steps, that would transform the otherwise abstract formula into an “inventive concept.” *See Mayo*, 132 S Ct. at 1299. As in *Flook*, the claims are drawn to a mathematical formula and merely append conventional activity, which cannot supply the “particular application” needed to impart patentability. *Mayo*, 132 S. Ct. at 1299 (citing *Flook*).

C. Inclusion of General Computer Components

Finally, the recitation that the method is to be used on conventional computer components does not make the abstract formula patentable. The parties’ constructions of the terms describing the physical components on which the patent method is practiced confirm that those structures are simply generic computer components. The physical components recited in the claims include “computer,” “computer storage,” and “z-buffers.” “Computer” was given no construction, while “computer storage” is agreed to mean simply “computer memory.” The term “z-buffer” was agreed to mean “a data structure in memory that is used to store depth data,” again a generic term. (Dkt. No. 64 at 6.)

The inclusion of these computer components does not impose a limitation sufficient to take the patents out of the realm of the abstract. As noted by Defendant, five of the claims here — claims 1, 4, and 5 of the ‘679 Patent and claims 1 and 12 of the ‘047 Patent— were the subject of prior litigation in which they were found to fail the machine-or-transformation test. *Fuzzysharp Techs., Inc. v. 3D Labs, Inc.*, 2009 WL 4899215, at *5 (N.D. Cal. Dec. 11, 2009). There the district

1 court, examining some of the exact claims at issue here, found that the claimed sequence of
2 “‘identifying,’ ‘comparing,’ ‘determining,’ and ‘ignoring’ data” “may be” performed on a computer
3 but does not require “any *particular* computer.” *Id.* at *4. Nor did such elements as a “method of
4 reducing the [] visibility related computations in 3-D graphics,” “computer storage,” “using a data
5 structure in a computer,” and “projecting 3D images ‘on a computer screen’” “impose any
6 meaningful limit on the claim scope” because it merely “serve[d] to perform the computation.” *Id.*
7 at *5 n.3. On appeal, the Federal Circuit found:

8 [t]he references to a computer in claim 12 impose only two limitations: the
9 machine must be able to compute, and it must be able to store data. Those
10 functions are essentially synonymous with the term “computer” and thus add
11 little or nothing to simply claiming the use of a general purpose computer. The
12 recitation of computer functions in the claim thus does not confine the
preemptive effect of the claim because the underlying method has “no
substantial practical application except in connection with a digital computer.”

13 *Fuzzysharp Technologies Inc. v. 3DLabs Inc., Ltd.*, 447 F. App'x 182, 185 (Fed. Cir. 2011), reh'g
14 denied (Dec. 20, 2011) (quoting *Benson*, 409 U.S. at 71). Thus the Federal Circuit agreed with the
15 district court’s conclusion that the claims were drawn to an abstract mathematical formula. *Id.* at
16 184. However, the district court had relied exclusively on the machine-or-transformation test,
17 which the Supreme Court subsequently held was not a definitive test of patent eligibility for a
18 method claims, but simply “a useful and important clue” to patent-eligibility. *Bilski*, 130 S. Ct. at
19 3227. Thus, the Federal Circuit vacated and remanded the judgment in the *3DLabs* action to permit
20 further consideration in light of *Bilski* and to allow for claim construction related to eligibility of
21 one unspecified claim there. *Id.* at 186.

22 The Court agrees with the determination in the *3DLabs* case that the computer components
23 included in the language of the Asserted Claims does not provide a meaningful limitation on the
24 abstract mathematical formula so as to render it patent eligible. The stated components are no more
25 specific than the computer components or computer-aided processing described repeatedly in the
26 case law as insignificant to patentability. As in *Benson*, *Accenture*, and similar authorities, the
27 specification of the method as requiring use of generic computer memory does not meaningfully
28 limit the broad concept set forth in the claims. See *Benson*, 409 U.S. at 67; *Accenture*, 728 F.3d at

1345-46. Adding a computer storage step to an otherwise unpatentable process, even where the claim names a particular type of storage component like the z-buffer here, does not transform it into a claim that is patent-eligible. *Cybersource*, 654 F.3d at 1375-76 (citing *Benson*, 409 U.S. at 73-74). Here, no claim recites any specific programming or computer component, but only geometric calculations. Performing those calculations on a computer, even if their only practical application is in a computing environment, does not render them patent-eligible. *Id.*

The eight additional Asserted Claims of the '047 Patent, not at issue in *3DLabs*, consist of the same method steps as the others (*i.e.*, identifying, comparing and determining visibility data, and then ignoring a set of data for certain surfaces in carrying out standard visibility calculations thereafter). As a result, those claims likewise suffer from the same lack of meaningful limitations on the mathematical formula at the core of the patent claims.

Unlike *Ultramercial*, the claims here suggest purely mathematical efficiencies, and do not require an "extensive computer interface" involving multiple steps and limitations. *Ultramercial*, 722 F.3d at 1352-53. Nor do the claims in the patents-in-suit implicate particular physical elements beyond a generic computer environment for carrying out their steps. *Cf. Research Corp. Technologies*, 627 F.3d at 869 (invention describing halftone rendering process in which digital data processor was utilized, along with "high contrast film," "film printer," as well as generic computer devices such as "memory" and "printer and display devices," was not so abstract as to be lose patent eligibility).

III. CONCLUSION

Based upon the foregoing, Intel's Motion for Summary Judgment for patent ineligibility of the Asserted Claims under Section 101 is **GRANTED**.

Intel is directed to submit a proposed form of judgment forthwith.

This Order terminates Docket No. 28.

IT IS SO ORDERED.

Date: November 6, 2013


YVONNE GONZALEZ ROGERS
UNITED STATES DISTRICT COURT JUDGE

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
OAKLAND DIVISION

FUZZYSHARP TECHNOLOGIES
INCORPORATED,

Plaintiff,

v.

INTEL CORPORATION,

Defendant.

Case No. 12-CV-4413 YGR

FINAL JUDGMENT

On August 22, 2012, Fuzzysharp Technologies Incorporated (“Fuzzysharp”) filed a Complaint (Dkt. No. 1) alleging that Intel Corporation (“Intel”) infringed U.S. Patent Nos. 6,172,679 and 6,618,047 (the “’679 patent” and “’047 patent,” respectively). On November 28, 2012, Intel filed an Answer and Counterclaims (Dkt. No. 16) that included counterclaims seeking declarations of non-infringement and invalidity of the patents-in-suit.

On March 15, 2013, Intel filed a Motion for Judgment on the Pleadings that the 40 claims then asserted by Fuzzysharp are invalid because they do not cover patent-eligible subject matter. Dkt. No. 28; *see also* Dkt. Nos. 32 and 33. On April 29, 2013, the Court converted that motion to a motion for summary judgment and directed the parties to identify any terms or phrases that required construction before a decision on summary judgment. Dkt. No. 45; *see also* Dkt. Nos. 47, 48, and 50. On August 9, 2013, Fuzzysharp withdrew all but claims 1, 4, and 5 of the ’679 patent and claims 1, 8, 11, 12, 13, 46, 57, 64, 65, and 67 of the ’047 patent (collectively, the “asserted claims”). Dkt. No. 66-2. On September 9, 2013, the parties submitted a Final Joint Claim Construction and Prehearing Statement that narrowed the claims construction issues to two terms. Dkt. No. 64-1.

On November 6, 2013, the Court entered an Order construing the two terms that remained in dispute and concluding that the asserted claims do not state eligible subject matter. Dkt. Nos. 74-76, “Order.” On December 5, 2013, the parties submitted a stipulation concerning the effect of that order on the pending claims and counterclaims, as well as the disposition of the remaining claims and counterclaims. Dkt. No. 82, “Stipulation.”

Pursuant to that Order and Stipulation, the Court now **ENTERS FINAL JUDGMENT** as follows:

Claims 1, 4, and 5 of the ’679 patent and claims 1, 8, 11, 12, 13, 46, 57, 64, 65, and 67 of the ’047 patent are invalid under 35 U.S.C. § 101.

Intel’s counterclaims for declarations of patent invalidity are **GRANTED** to that extent.

Intel’s other counterclaims for declarations for patent invalidity are **DISMISSED WITHOUT PREJUDICE AS MOOT.**

1
2
3 Additionally, the Court now **ENTERS FINAL JUDGMENT** that Intel is not liable for
4 infringement, if any, of the asserted claims. Intel's counterclaims for declaratory judgment of
5 noninfringement are **DISMISSED WITHOUT PREJUDICE AS MOOT**.

6 This judgment thus disposes of all claims and counterclaims before the Court.

7 This is a final, appealable judgment.

8 **IT IS SO ORDERED.**

9 Dated: December 11, 2013

10 

11 Honorable Yvonne Gonzalez Rogers
12 UNITED STATES DISTRICT COURT JUDGE
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

(12) **United States Patent**
Lim

(10) **Patent No.:** **US 6,618,047 B1**
(45) **Date of Patent:** ***Sep. 9, 2003**

(54) **VISIBILITY CALCULATIONS FOR 3D COMPUTER GRAPHICS**

(75) Inventor: **Hong Lip Lim**, Singapore (SG)

(73) Assignee: **Fuzzysharp Technologies, Inc.**, Las Vegas, NV (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **09/490,288**

(22) Filed: **Jan. 20, 2000**

Related U.S. Application Data

(63) Continuation of application No. 08/935,150, filed on Sep. 22, 1997, now Pat. No. 6,172,679, which is a continuation of application No. 08/182,096, filed as application No. PCT/AU92/00302 on Jun. 19, 1992, now Pat. No. 5,914,721.

(30) Foreign Application Priority Data		
Oct. 1, 1991	(AU)	PK8643
Oct. 1, 1991	(AU)	PK8645
Oct. 30, 1991	(AU)	PK9218
Jul. 19, 1991	(AU)	PK7305
Jun. 28, 1991	(AU)	PK6942

(51) **Int. Cl.⁷** **G06T 15/40**

(52) **U.S. Cl.** **345/421; 345/422**

(58) **Field of Search** **345/419, 421, 345/425, 114, 139, 422, 418, 424**

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,594,673	A *	6/1986	Holly	345/421
4,625,289	A *	11/1986	Rockwood	345/422
4,697,178	A	9/1987	Heckel	
4,819,192	A	4/1989	Kuragano et al.	
4,825,391	A *	4/1989	Merz	
4,855,938	A *	8/1989	Gonzalez-Lopez et al.	345/422

(List continued on next page.)

FOREIGN PATENT DOCUMENTS

EP	193151	A	9/1986
EP	0 481 581		4/1992
GB	2 228 850		9/1990

OTHER PUBLICATIONS

“Stabbing Isothetic Boxes & Rectangles in $O(n \lg n)$ time”, M. Hohmeyer S. J. Teller, 1991.*

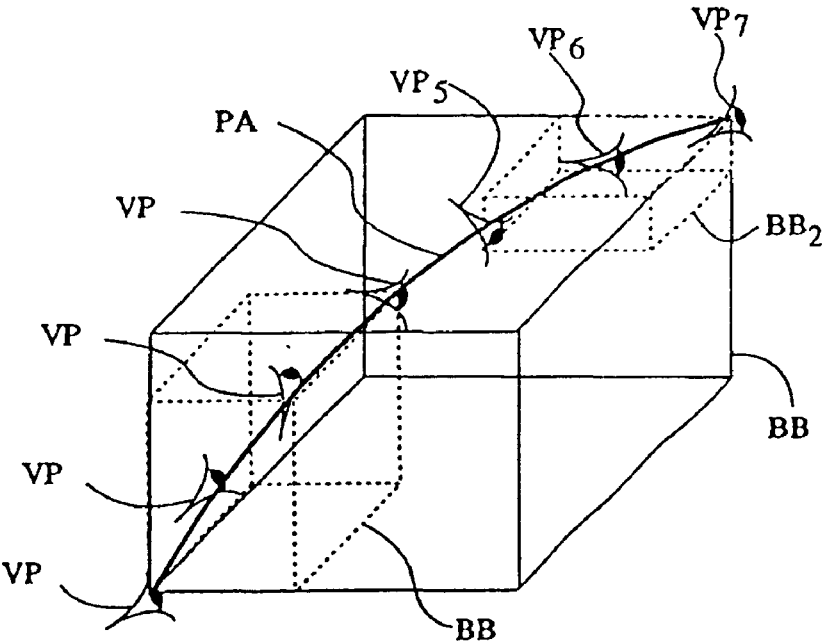
(List continued on next page.)

Primary Examiner—Jeffery Brier
(74) *Attorney, Agent, or Firm*—Fish & Richardson P.C.

(57) **ABSTRACT**

Disclosed is a method of reducing the complexity of hidden surface removal in 3D graphics systems. A fuzzy projection (FF) of a surface (SU) as seen from a number of viewpoints (VP) in a bounding box (BB) is stored in a buffer (FA) having elements (FE). A combination of all the patches (PT) of the surface (SU) viewed from a fuzzy region (FR) where surfaces can be either visible, hidden, or unable to be determined with certainty as to whether or not visible/hidden. A non-fuzzy region (NF) describes those patches (PT) that are always visible.

68 Claims, 16 Drawing Sheets



US 6,618,047 B1

Page 2

U.S. PATENT DOCUMENTS

4,901,252	A	2/1990	Fitzgerald et al.	
4,918,626	A *	4/1990	Watkins et al.	345/421
4,928,250	A *	5/1990	Greenberg et al.	345/426
5,027,292	A *	6/1991	Rossignac et al.	345/422
5,058,042	A *	10/1991	Hanna et al.	345/427
5,081,698	A	1/1992	Kohn	
5,084,830	A *	1/1992	Doornink et al.	
5,086,496	A	2/1992	Mulmuley	
5,088,054	A	2/1992	Paris, II	
5,159,663	A	10/1992	Wake	345/422
5,177,474	A *	1/1993	Kadota	345/139
5,253,335	A	10/1993	Mochizuki et al.	345/422
5,268,996	A	12/1993	Steiner et al.	345/426
5,295,243	A	3/1994	Robertson et al.	
5,299,298	A	3/1994	Elmqvist et al.	345/421
5,313,568	A	5/1994	Wallace et al.	345/426
5,329,613	A *	7/1994	Brase et al.	345/422
5,377,313	A	12/1994	Scheibl	
5,384,580	A *	1/1995	Kadota	345/419
5,402,532	A	3/1995	Epstein et al.	
5,414,801	A	5/1995	Smith et al.	
5,448,686	A	9/1995	Borrel et al.	
5,619,592	A	4/1997	Bloomberg et al.	
5,619,593	A	4/1997	Ono	
5,644,689	A *	7/1997	Ban et al.	345/424
5,914,721	A *	6/1999	Lim	345/421
6,172,679	B1	1/2001	Lim	

OTHER PUBLICATIONS

"A Characterization of Ten Hidden-Surface Algorithms", I. E. Sutherland, R. F. Sproull, R. A. Schumacker Computing Surveys, vol. 6, No. 1, 1975.*

"Fast Algorithms for 3D-Graphics", G. Glaeser, 1994.*

**Gigus et al., "Efficiently computing and representing aspect graphs of polyhedral objects", 2nd International conference on computer vision, 1988.

**Newman et al., "Principles of interactive computer graphics", 2nd ed. 1979, McGraw Hill.

Accelerated radiosity method for complex environments H. Xu Q. Peng, Y. Liang Eurographics'89 1989.

Application Challenges to Computational Geometry, CG Impact Task Force Report CG Impact Task Force Technical Report TR-521-96, Princeton University 1996.

Computer Graphics, Principles and Practice, 2nd edition J. D. Foley A. van Dam S. K. Feiner J. F. Hughes.

New Trends in Animation and Visualization N. Thalmann D. Thalmann 1991.

Graphics Systems: Architecture & Realization R. Andreev 1993.

Analysis of Radiosity Techniques in Computer Graphics B. Kwok MSc Thesis York University, May 1992.

Image display data computer forming method—uses perspective transformation with calculation time reduction on shadow and hidden surface processing Sony 86.233751/36.

Stabbing and ray shooting in 3 dimensional space M. Pellegrini 1990.

The Geometry of Beam Tracing N. Dadoun D. Kirkpatrick 1985.

Algorithms for line transversals in space D. Avis CG1987.

Optimization of the binary space partition algorithm (BSP) for the visualization of dynamic scenes E. Torres Eurographics'90.

A mathematical Semantics of Rendering II. Approximation E. Fiume CVGIP: Graphical Models and Image Processing vol. 53, No. 1, Jan. 1991.

A Characterization of Ten Rasterization Techniques N. Gharachorloo S. Gupta R. Sproull I. Sutherland Computer Graphics, vol. 23, No. 3 (Siggraph'89) 1989.

The use of projective geometry in Computer Graphics I. Herman 1992.

Ray Tracing with Cones J. Amanatides Computer Graphics, vol. 18, No. 3 (Siggraph'84) 1984.

The A-buffer, an Antialiased hidden surface method L. Carpenter Computer Graphics, vol. 18, No. 3, (Siggraph'84) 1984.

Principles and Applications of Pencil Tracing M. Shinya T. Takahashi S. Naito Computer Graphics, vol. 21, No. 4, (Siggraph'87) 1987.

Light-water interaction using backward beam tracing M. Watt Computer Graphics, vol. 24, No. 4 (Siggraph'90) 1990.

Rendering CSG Model with a ZZ-buffer D. Salesin J. Stolfi Computer Graphics, vol. 24, No. 4 (Siggraph'90) 1990.

A solution to the hidden-line problem for computer-drawn polyhedra P. Loutrel IEEE Transactions on Computers, Mar. 1970.

Sorting and the hidden-surface problem I. Sutherland R. Sproull R. Schumacker National Computer Conference, 1970.

The Notion of quantitative invisibility and the machine rendering of solids A Appel ACM National Meeting 1967.

An analytic visible surface algorithm for independent pixel processing E. Catmull Computer Graphics, vol. 18. No. 3 (Siggraph'84) 1984.

Computing the lines piercing four lines S. Teller M. Hohmeyer Technical Report 92-665, University of California, Berkeley.

Computing the antipenumbra of an area light source S. Teller Computer Graphics, vol. 26, No. 2, (Siggraph'92) 1992.

Computer animation, theory and practice, second revised edition N. Thalmann D. Thalmann 1990.

Image Synthesis M. Brest 1992.

Management of large amounts of data in interactive building walkthroughs T. Funkhouser C. Sequin S. Teller Symposium on Interactive 3D Graphics, 1992.

Temporal Coherence in Ray Tracing S. H. Badt PhD thesis, University of Texas at Dallas, 1989.

Near real-time shadow generation using BSP trees N. Chin S. Feiner Computer Graphics, vol. 23, No. 3 (Siggraph'89) 1989.

Adaptive Display algorithm for interactive frame rates during visualization of complex virtual environments T. Funkhouser C. H. Sequin Siggraph'93 1993.

Modeling global diffuse illumination for image synthesis A. Campbell, III. PhD Thesis University of Texas at Austin 1991.

A 3-dimensional representation for fast rendering of complex scences S. Rubin T. Whitted 1980.

Radiosity redistribution for dynamic environment D. George F. Sillion D. Greenberg IEEE Computer Graphics & Applications, vol. 4, 1990.

A survey of shadow algorithms A. Woo, F. Poulin A. Fournier IEEE Computer Graphics & Applications, vol. 6, 1990.

Error-bounded antialiased rendering of complex environments N. Greene M. Kass Siggraph'94 1994.

Fast computation of shadow boundaries using spatial coherence and backprojections A. Stewart S. Ghali Siggraph'94 1994.

US 6,618,047 B1

Page 3

- A fast shadow algorithm for area light sources using back-projection G. Drettakis E. Fiume Siggraph'94 1994.
- Stabbing oriented convex polygons in randomized $O(n^2)$ time. S. Teller M. Hohmeyer Contemporary Mathematics, 1994.
- Obscuration culling on parallel graphics architecture C. George Technical report TR95-017 University of North Carolina at Chapel Hill 1995.
- Visibility between two edges of a simple polygon D. Avis T. Gum G. Goussaint The Visual Computer, 1986, No. 2.
- Increasing update rates in the building walkthrough system with automatic model-space subdivision and potentially visible set calculations J. M. Airey PhD Thesis, University of North Carolina, 1990.
- Realism in computer graphics: a survey. J. Amanatides IEEE Computer Graphics and Applications, vol. 7, No. 1, 1987.
- Finding a line transversal of axis objects in three dimensions N. Amenta Proc. 3rd Annual ACM-SIAM Symposium on Discrete Algorithms, 1992.
- Beyond the third dimension, geometry, computer graphics and higher dimension Banchoff T. F. Scientific American Library 1990.
- A general version of crow's shadow volumes Bergeron P. IEEE Computer Graphics and Applications, vol. 6, No. 9, Sep. 1986.
- Me and my (fake shadow) Blinn, J. F. IEEE Computer Graphics and Applications, vol. 8, No. 1 1988.
- Generating soft shadows with a depth buffer algorithm Brotman L. S. IEEE Computer Graphics and Applications vol. 4, No. 10, 1984.
- A multiresolution spline with application to image mosaics Burt P. J. Adelson E. H. ACM Transactions on Graphics, vol. 2, Oct. 1983.
- Hierarchical geometric models for visible surface algorithms Clark, J. H. Communications of the ACM, vol. 19, No. 10, Oct. 1976.
- An overview of rendering techniques Dennis A. R. Computer & Graphics, vol. 14, No. 1, 1990.
- Hybrid shadow testing scheme for ray tracing Eo K. S. Kyung C. M. Computer Aided Design vol. 21, No. 1 Jan. 1989.
- Hierarchical rendering of complex environments Greene, N. PhD dissertation, University of California, Santa Cruz 1995.
- Bibliography of hidden-line and hidden-surface algorithms Griffiths, J. G. Computer-Aided Design, vol. 10, May 1978.
- The light buffer: a shadow-testing accelerator Haines, E. A. Greenberg, D. IEEE Computer Graphics and Applications, vol. 6, No. 9, Sep. 1986.
- Algorithms for antialiased cast shadows Hourcade J. C. Nicolas A. Computer and Graphics, vol. 9, No. 3 1985.
- Hemi-cube ray-tracing: a method for generating soft shadows Meyer U. Proceedings, Eurographics'90, 1990.
- Principles of interactive computer graphics, 1st edition Newman, W. M. Sproull, R. F. 1973.
- A comparison of four visibility acceleration techniques for radiosity Ng, A. The visual computer, 1996, vol. 12, pp. 307-316.
- Shading models for point and linear sources Nishita, T. Okamura, I. Nakamae, E. ACM transactions on graphics, vol. 4, No. 2, Apr. 1985.
- Radiosity of dynamic scenes in flatland with the visibility complex Orti, R. Riviere S. Durand F. Puech C. Proceedings, Eurographics'96 1996.
- Visibility, occlusion, and the aspect graph Plantinga H. Dyer C. R. International Journal of computer vision, vol. 5, No. 2, 1990.
- Necessary and sufficient conditions for hyperplane transversal Pollack R. Wenger R. Proc. 5th Annual Symposium on Computational Geometry, 1989.
- Shading and shadowing with linear light sources Poulin P. Amanatides J. Proceedings, Eurographics'90 1990.
- Rendering antialiased shadows with depth maps Reeves W. Salesin D. Cook R. Proceedings, Siggraph'87 1987.
- Machine perception of three-dimensional solids Roberts L. G. Optical and Electro-Optical Information Processing, J. Tippet (editor) MIT Press, 1965.
- Procedural elements for computer graphics Rogers D. F. McGraw-Hill (publisher) 1985.
- Optics B. Rossi Addison-Wesley (publisher) 1957.
- An optimal algorithm for detecting weak visibility of a polygon J. Sack S. Suri IEEE transactions on computers, vol. 39, No. 10, Oct. 1990.
- Shaded rendering and shadow computation for polyhedral animation Seales W. B. Dyer C. R. Proceedings, Graphics Interface '90, May 1990.
- Optics, third edition Sears F. W. Addison-Wesley Publishing Co. 1949.
- Linear programming and convex hulls made easy Seidel R. Proc. 6th ACM Symposium on Computational Geometry, 1990.
- Output-sensitive visibility algorithms for dynamic scenes with applications to virtual reality Sudarsky O. Gotsman C. Proceedings, Eurographics'96 1996.
- Automatic view function generation for walk-through animation using a reeb graph Shinagawa Y. kunii T. Nomura Y. Okuna T. Young Y. Computer Animation '90 1990.
- Octant priority for radiosity image rendering Wang Y. Davis W. Proceedings, Graphics Interface '90 1990.
- Casting curved shadows on curved surfaces Williams L. Proceedings, Siggraph'78 1978.
- Pyramidal parametrics William L. Proceedings, Siggraph'83 1983.
- Accelerated radiosity method for complex environments Xu H. Peng Q. Liang Y. Proceedings, Eurographics'89 1989.
- Pyramid clipping for efficient ray traversal Zwann M. Reinhard E. Jansen F. Proceedings of the Sixth Eurographics Rendering Workshop 1995.
- Application challenges to computational geometry, CG impact task force report CG impact task force Princeton University Computer Science Dept. Technical Report TR-521-96 <http://graphics.lcs.mit.edu/about.seth/pubs/taskforce/techrep.html> <http://www.cs.princeton.edu/about.chazelle/taskforce/CGreport.ps.Z> <http://graphics.lcs.mit.edu/about.seth/pubs/Pubs.html> 1996.
- My response to application challenges to computational geometry Franklin R. <http://www.ecse.rpi.edu/Homeworks/wrf/geom.sub.—response.html> <http://netlib.bell-labs.com/netlib/compgeom/discuss/archive/96/ta skforce.html> 1996.
- Comments on the report "Application challenges to computational geometry" Heckbert P. <http://www.cs.duke.edu/about.jeffe/compgeom/files/heckbert.html> <http://netlib.bell-labs.com/netlib/compgeom/discuss/archive/96/ta skforce.html> 1996.

US 6,618,047 B1

Page 4

- Follow-up comments on the report "Application challenges to computational geometry" Coulson T. <http://www.cs.duke.edu/~about.jeffe/compgeom/files/coulson.html> <http://netlib.bell-labs.com/netlib/compgeom/discuss/archive/96/ta skforce.html> 1996.
- Inside Quake: visible surface determination <http://www.gamers.org/dEngine/quake/papers/ddjpvs.html> 1996.
- CGDC Quake Talk <http://www.gamers.org/dEngine/quake/papers/mikeab-cgdc.html> 1996.
- Quake hidden surface removal <http://www.gamers.org/dEngine/quake/papers/ddjzsort.html> 1996.
- Quake editing tools information <http://www.gamers.org/dEngine/quake/QuakeEd/qedit.sub.—infor.html> 1996.
- Zen of graphics programming Abrash M. 1996.
- Computer graphics: more unsolved problems Siggraph'91 panel 1991.
- Global illumination in architecture and entertainment Siggraph'96 course notes 1996.
- Interactive walkthrough of large geometric databases Siggraph'96 course notes 1996.
- Imprecise computation and load sharing in computer generated imaging system Berger M. Zhao W. Graphics Interface '90 1990.
- Exploiting temporal coherence in ray tracing Chapman J. Calvert T. Sill J. Graphics Interface'90 1990.
- Approximate ray tracing Dauenhauer D. Graphics Interface '90 1990.
- Approximate and probabilistic algorithms for shading and rendering structured particle systems Reeves W. Siggraph 1985 1985.
- Fundamentals of interactive computer graphics Foley J.D. van Dam A. Addison-Wesley Publishing Co. 1982.
- Mult-pass multi-resolution algorithm for the determination of hidden surfaces Lim, H. L. Inter-faculty symposium on computer graphics and image processing 1987.
- Ten Unsolved Problems in Rendering Heckbert P. S. Workshop on Rendering Algorithms and Systems, Graphics Interface'87 [www addr: http://www.cs.cmu.edu/about.ph](http://www.cs.cmu.edu/about.ph) (index page) 1987.
- Cainter's & z-buffer algorithms Course note, CS dept, Carnegie-Melon university [www addr: http://www.cs.cmu.edu/afs/cs/project/anim/ph/463.95/pub/www/notes.t-oc.html](http://www.cs.cmu.edu/afs/cs/project/anim/ph/463.95/pub/www/notes.t-oc.html) 1996.
- 3D comes alive Ozer, J. PC magazine, Jun. 25, 1996.
- Affordable 3-D workstations Hummel, R. L. Byte Dec. 1996.
- Gaming in the next dimension Case, L. Salvator, D. Computer Gaming Jul. 1996.
- 3D engine list Isakovic, K. [www addr: http://www.cs.tu-berlin.de/~about.ki/engines.html](http://www.cs.tu-berlin.de/~about.ki/engines.html) 1996.
- OpenGL the leading visual programming interface [www addr: http://www.sgi.com/Products/Dev.sub.—environ.sub.—ds.html](http://www.sgi.com/Products/Dev.sub.—environ.sub.—ds.html) 1996.
- 3D computer graphics (second edition) Glassner A. S. Design Press 1989.
- The UC Berkeley system for interactive visualization of large architectural models T. Funkhouser S. Teller C. Sequin D. Khorramabadi Presence, vol. 5, No. 1, Winter 1996.
- Visibility computation for efficient walkthrough of complex environment R. Yagel R. William Presence, vol. 5, No. 1, Winter 1996.
- Large models for virtual environments: A review of work by the architectural walkthrough project at UNC M. R. Mine H. Weber Presence, vol. 5, No. 1, Winter 1996.
- A hidden-line algorithm for hyperspace R. P. Burton D. R. Smith SIAM Journal of Computing vol. 11, No. 1, Feb. 1982.
- Canonic representations for the geometrics of multiple projective views Q. T. Luong T. Vieville University of California at Berkely CS Dept. Tech. Report UCB/CSD093-772 1993.
- Multidimensional graphing in two-dimensional spaces T. Mihalisin E. Gawlinski J. Timlin J. Schwegler Computers in Physics, Nov./Dec. 1989.
- A framework for global illumination in animated environments J. Mineroff J. Dorsey H. Rushmeier Rendering Techniques'95 (Proceedings of the Eurographics Workshop in Dublin, Ireland Jun. 1995).
- Shadows for bump-mapped surfaces N. L. Max Advanced Computer Graphics (Proceedings of Computer Graphics Tokyo '86) 1986.
- The simulatoin of natural features using cone tracing D. Kirk Advanced Computer Graphics (Proceedings of Computer Graphics Tokyo '86) 1986.
- Simulating soft shadows with graphics hardware P. S. Heckbert M. Herf Carnegie-Melon University CS Dept. Technical Report CMU-CS-97-104 1997.
- Spatial transformation for rapid scan-line surface shadowing P. Robertson IEEE computer graphics & applications Mar. 1989.
- Incremental update of the visibility map as seen by a moving viewpoint in two dimensions S. Ghali A. J. Stewart Eurographics Workshop on Animation and Simulation, Aug. 1996.
- Z. H. Zhao D. Dobkin Continuous algorithms for visibility: the space searching approach fourth eurographics workshop on rendering 1993.
- Y. Shinagawa S. Miyoshi T. Kunii Viewpoint analysis of drawings and paintings rendered using multiple viewpoints: cases containing rectangular objects Fourth Eurographics Workshop on Rendering 1993.
- F. Jansen A. Chalmers Realism is real time Fourth Eurographics Workshop on Rendering 1993.
- P. Schuytema How to bake a quake Computer Gaming Jul. 1996.
- Efficient Collision Detection for Animation and Robotics Ming C. Lin 1993.
- Fast Computation of Shadow Boundaries Using Spatial Coherence and Backprojections A James Stewart Sherif Ghali.
- Polygon shadow generation Atherton P. Weiller K. Greenberg D. Proceedings, Siggraph'78.
- Designing real-time 3d graphics for entertainment Siggraph'96 course notes.
- Foley, "Computer Graphics: Principles and Practice, Second Edition" Addison-Wesley Publishing Company, Inc. pp. 663-665, 772, 783, 793-797, 805 (1990).
- Sutherland, "A Characterization of Ten Hidden-Surface Algorithms," Computing Surveys, vol. 6, No. 1, pp. 38-39, Mar. 74.
- Jain et al., "Imprecision in Computer Vision", Advances in Fuzzy Sets, Possibility and Applications, 1983.
- E. Catmull, "Computer display of curved surfaces" in IEEE Transactions of Computers, p. 309-315, 1971.
- M. F. Cohen et al., "The hemi-cube: A Radiosity solution for complex environment", Computer Graphics (SIGGRAPH '85), vol. 19, No. 3, p. 31-40, 1985.

US 6,618,047 B1

Page 5

- G.A. Crocker, "Invisibility coherence for faster scan-line hidden surface algorithms", *Computer Graphics*, vol. 18, No. 3, p. 95-102, 1984.
- H. Hubschman et al., "Frame-to-frame coherence and the hidden surface computations: Constraints for a Convex World", *Computer Graphics*, vol. 15, No. 3, p. 45-54, 1981.
- H.L. Lim, "Fast hidden surface removal through structural analysis and representation of objects and their contours", *Computer Graphics International '87*, p. 75-88, 1987.
- H.L. Lim, "Toward a fuzzy hidden surface algorithm", *Computer Graphics International '92*, p. 621-635, 1992.
- H.L. Lim, "An efficient hidden surface algorithm for polyhedral surfaces" in *International Conference on Computer & Communications in Science & Technology*, Beijing, China, 1986.
- C. Hornung, "A method for solving the visibility problem", *IEEE Computer Graphics & Applications*, vol. 4, p. 26-33, 1984.
- J. Griffiths, "A depth-coherence scanline algorithm for displaying curved surfaces", *Computer-aided Design*, vol. 16, No. 2, p. 91-101, 1984.
- E.A. Haines et al., "Shaft culling for efficient ray-traced radiosity", *Proceedings of Eurographics Workshop on Rendering*, Jul., 1991.
- H. Plantinga et al., "Real-time hidden-line elimination for a rotating polyhedral scene using the aspect representation", in *Graphics Interface '90 Proceedings*, p. 9-16, 1990.
- J.M. Airey et al., "Towards image realism with interactive update rates in complex virtual building environment", in *ACM SIGGRAPH Special Issue on the 1990 Symposium on Interactive 3D Graphics* vol. 24, p. 41-50, 1990.
- Y. Wang, "Image synthesis using front-to-back based radiosity methods", Ph.D. thesis, University of Alberta, 1992.
- D.R. Baum et al., "The back-buffer algorithm: an extension of the radiosity method to dynamic environments", *The Visual Computer*, vol. 2, No. 5, p. 298-306, 1986.
- S.J. Teller et al., "Visibility preprocessing for interactive walkthroughs", *Computer Graphics*, vol. 25, No. 4, p. 61-69, 1991.
- J. Marks et al., "Image and intervisibility coherence in rendering", *Graphics Interface '90 Proceedings*, p. 17-30, 1990.
- H.L. Lim, "Rendering techniques in three-dimensional computer graphics", Ph.D. thesis, University of Sydney, 1993.
- S.J. Teller, "Visibility computations in densely occluded polyhedral environment", Ph.D. thesis, University of California at Berkeley, 1992.
- A.S. Glassner, "Principles of Digital Image Synthesis", vol. 2, Morgan Kaufmann Publisher, San Francisco, 1995.
- D. Gordon et al., "Front-to-back display of BSP trees", *IEEE Computer Graphics & Applications*, vol. 11, p. 79-85, 1991.
- K.L. Shelley et al., "Path specification and path coherence", *Computer Graphics*, vol. 16, No. 3, p. 157-161, 1982.
- J. Vilaplana et al., "Exploiting coherence for clipping and view transformations in radiosity algorithms", in *Eurographics Workshop on Photosimulation, Realism and Physics in Computer Graphics*, Rennes, France, p. 137-149, 1989.
- C.B. Jones, "A new approach to the 'hidden line' problem", *Computer Journal*, vol. 14, No. 3, p. 232-236, 1971.
- F.C. Crow, "Shadow algorithms for computer graphics", *Computer Graphics*, vol. 11, No. 2, p. 442-448, 1977.
- X. Pueyo, "The use of visibility coherence for radiosity computation", in *First International Conference on Visualization and Intelligent Design in Engineering and Architecture*, p. 17-28, 1993.
- T. Nishita et al., "Continuous tone representation of three-dimensional objects taking account of shadows and interreflection", *Computer Graphics*, vol. 19, No. 3, p. 23-30, 1985.
- T. Funkhouser, "Database and display algorithms for interactive visualization of architectural models", Ph.D. thesis, University of California at Berkeley, 1993.
- S.J. Teller et al., "Global visibility algorithms for illumination computations", in *Computer Graphics (SIGGRAPH '93)*, vol. 27, p. 239-246, 1993.
- S. Coorg et al., "Temporally coherent conservative visibility", in *Twelfth Annual ACM Symposium on Computational Geometry*, Philadelphia, ACM Press, New York, p. 1-10, 1996.
- S. Coorg et al., "A spatially and temporally coherent object space visibility algorithm", *Laboratory of Computer Science, MIT, Technical Report TM-546*, 1996.
- D.P. Luebke et al., "Portal and mirrors: simple, fast evaluation of potentially visible sets", in *Proceedings 1995 Symposium on Interactive 3-D Graphics*, ACM Press, New York, 1995.
- H. Plantinga, "Conservative visibility preprocessing for efficient walkthroughs of 3D scenes", *Graphics Interface '93 Proceedings*, p. 166-173, 1993.
- E. Catmull, "A subdivision algorithm for computer display of curved surfaces", Ph.D. thesis, Utah University, Dec. 1974.
- J. Arvo et al., "A survey of ray tracing acceleration techniques" in *An Introduction to Ray Tracing*, editor: A.S. Glassner, Academic Press, London, p. 201-262, 1989.
- A. Watt et al., "Advanced Animation and Rendering Techniques, Theory and Practice", ACM Press, New York, 1992.
- F. Sillion et al., "Radiosity and Global Illumination", Morgan Kaufmann Publisher, San Francisco, 1994.
- D.R. Baum et al., "Improving radiosity solutions through the use of analytically determined form-factors", *Computer Graphics*, vol. 23, No. 3, p. 325-334, 1989.
- A. Fournier et al., "On the power of the frame buffer", *ACM Transaction on Graphics*, vol. 7, No. 2, 103-128, 1988.
- C.W. Grant, "Integrated analytic spatial & temporal anti-aliasing for polyhedra in 4-Space", *Computer Graphics*, vol. 19, No. 3, p. 79-84, 1985.
- P. Hsiung et al., "T-Buffer: fast visualization of relativistic effects in spacetime", *ACM SIGGRAPH Special Issue on the 1990 Symposium on Interactive 3D Graphics* 24, p. 83-88, 1990.
- A. Inselberg, "The Plane with parallel coordinates", *The Visual Computer*, vol. 1, p. 69-91, 1985.
- N.L. Max et al., "A two-and-a-half-D motion-blur algorithm", *Computer Graphics (SIGGRAPH '85)*, vol. 19, No. 3, p. 85-93, 1985.
- K.V. Steiner et al., "Hidden volumes: the 4th dimension", *Computer Graphics World*, p. 71-74, Feb. 1987.
- L.A. Zadeh, "Fuzzy Sets", *Information and Control*, vol. 8, p. 338-353, 1965.
- W. Siedlecki et al., "Mapping techniques for exploratory pattern analysis" in *Pattern Recognition and Artificial Intelligence*, E.S. Gelsema, L.N. Kanal (eds), Elsevier, New York, p. 277-299, 1988.

US 6,618,047 B1

Page 6

- S. Chang, "On fuzzy mapping and control", IEEE Transactions on Systems, Man & Cybernetics, vol. SMC-2, No. 1, p. 30-34, 1972.
- R. Jain et al., "Imprecision in computer vision" in Advances in Fuzzy Sets, Possibility and Applications, P. Wang, (ed), Plenum Press, New York, p. 217-236, 1983.
- N. Greene et al., "Hierarchical & buffer visibility", Computer Graphics (SIGGRAPH '93), vol. 27, p. 231-238, 1993.
- D. Greenberg et al., "Radiosity: a method for computing global illumination", The Visual Computer, vol. 2, p. 291-297, 1986.
- H. Fuchs et al., "New real-time shaded display of rigid objects", Computer Graphics, Vol. 17, No. 3, p. 65-72, 1983.
- C.M. Hoffman et al., "Some techniques for visualizing surfaces in four-dimensional space", Computer-aided Design, vol. 23, No. 1, p. 83-91, 1991.
- J.M. Lane et al., "Scan line methods for displaying parametrically defined surfaces", Communications of the ACM, vol. 23, No. 1, p. 23-34, 1980.
- J.A. Gualtieri et al., "The visual potential: one convex polygon", Computer Vision, Graphics and Image Processing, vol. 46, No. 1, p. 96-130, 1989.
- Z. Gigus et al., "Computing the aspect graph for line drawings of polyhedral objects", in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Computer Society Press, New York, p. 654-661, 1988.
- P.S. Heckbert et al., "Beam tracing polygonal objects", Computer Graphics, vol. 18, No. 3, p. 119-127, 1984.
- J. Zhou, "Visualization of four dimensional space and its applications", Ph.D. thesis, Purdue University, 1991.
- R. Jain, "Application of fuzzy sets for the analysis of complex scenes" in Advances in Fuzzy Set Theory and Applications, M.M. Gupta et al. (eds), North-Holland, Amsterdam, p. 577-583, 1979.
- D. Tost et al., "A definition of frame-to-frame coherence", Computer Animation '90, p. 207-221, 1990.
- Y. Leung, "Spatial Analysis and Planning under Imprecision", North Holland, Amsterdam, 1988.
- J.R. Wallace et al., "A ray tracing algorithm for progressive radiosity", Computer Graphics, vol. 23, No. 3, p. 315-324, 1989.
- H. Fuchs et al., "On visible surface generation by a priori tree structure", Computer Graphics (SIGGRAPH'80), vol. 14, p. 124-133, 1980.
- M.F. Cohn et al., "A progressive refinement approach to fast radiosity image generation", Computer Graphics, vol. 22, No. 4, p. 75-84, 1988.
- K. Kanatani, "Group-theoretical Methods in Image Understanding", Springer Verlag, Berlin, 1990.
- J.K. Aggarwal et al., "Dynamic scene analysis" in Image Sequence Processing and Dynamic Scene Analysis, T.S. Huang (ed), Springer Verlag, Berlin, p. 40-73, 1983.
- N.I. Badler et al., "Motion: Representation and Perception" Elsevier, New York, 1986.
- Subbarao, "Interpretation of visual motion: A computational study", Pitman, London, 1988.
- F. Sillion et al., "A general two-pass method integrating specular and diffuse reflection", Computer Graphics, vol. 23, No. 3, p. 335-344, 1989.
- R.J. Recker et al., "Acceleration Techniques for Progressive Refinement Radiosity", ACM SIGGRAPH Special Issue on the 1990 Symposium on Interactive 3D Graphics, vol. 24, p. 59-66, 1990.
- E. H. Ruspini, "A new approach to clustering", Information and Control 15, p. 22-32, 1969.
- A. Kaufmann, "Theory of Fuzzy Subsets. Vol I, Fundamental Theoretical Elements", Academic Press, London, 1975.
- T.S. Huang, "Image Sequence Processing", Springer Verlag, Berlin, 1981.
- P.A. Ligomenides, "Modeling uncertainty in human perception" in Uncertainty in Knowledge-Based Systems, B. Bouchon, R. Yager (eds), Springer Verlag, Berlin, p. 337-346, 1986.
- A. Inselberg, "N-Dimensional Graphics. Part I. Lines & Hyperplanes", IBM Scientific Center Report G320-2711, Jul. 1981.
- A.S. Glassner, "3D Computer Graphics: A User's Guide for Artists & Designers", 2nd edition, Design Press, New York, p. 139-158, 1989.
- M.F. Cohen et al., "Radiosity & realistic image synthesis", Academic, New York, 1993.
- J. Aggarwal et al., "Analysing dynamic scenes containing multiple moving objects" in Image Sequence Analysis, editor: T.S. Huang, Springer-Verlag, Berlin, p. 355-380, 1981.
- A. Appel, "The Notion of quantitative invisibility and the machine rendering of solids", Proceedings ACM National Conference, Thompson Books, Washington, DC, p. 387-393, 1967.
- J. Vince, "Computer Animation", Addison-Wesley, New York, 1992.
- Y. Chrysanthou et al., "Computing dynamic changes to BSP-trees", Eurographics '92, vol. 11, No. 3, p. C-321-C-332, 1992.
- S. Ansoldi et al., "Geometric modeling of solid objects by using a face adjacency graph presentation" Computer Graphics (SIGGRAPH '85), vol. 19, No. 3, p. 131-138, 1985.
- N. Chin et al., "Fast object-precision shadow generation for area light sources using BSP trees", Proceedings 1992 Symposium on Interactive 3D Graphics, p. 21-30, 1992.
- D.S. Immel et al., "A radiosity method for non-diffuse environments", Computer Graphics, vol. 4, p. 133-142, 1986.

* cited by examiner

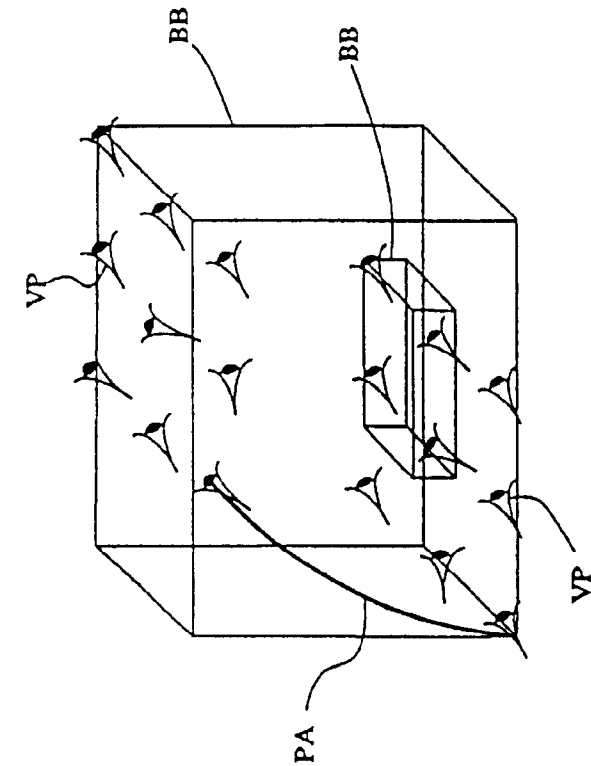


FIG. 1B

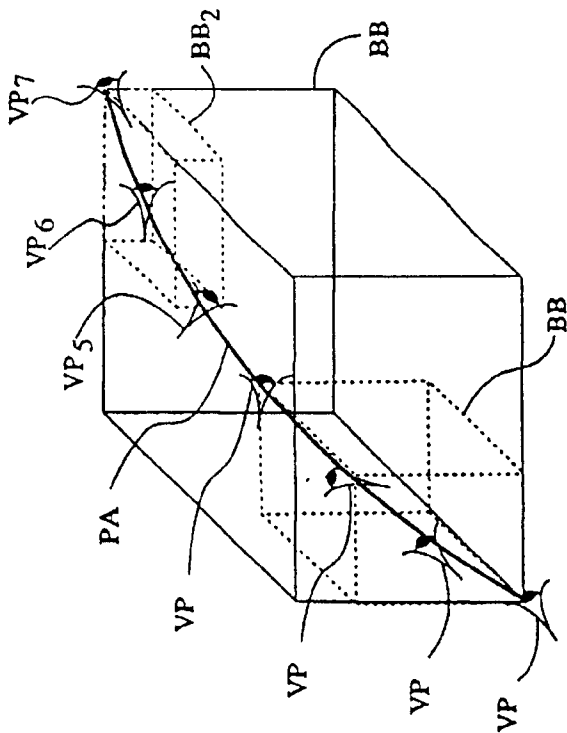


FIG. 1A

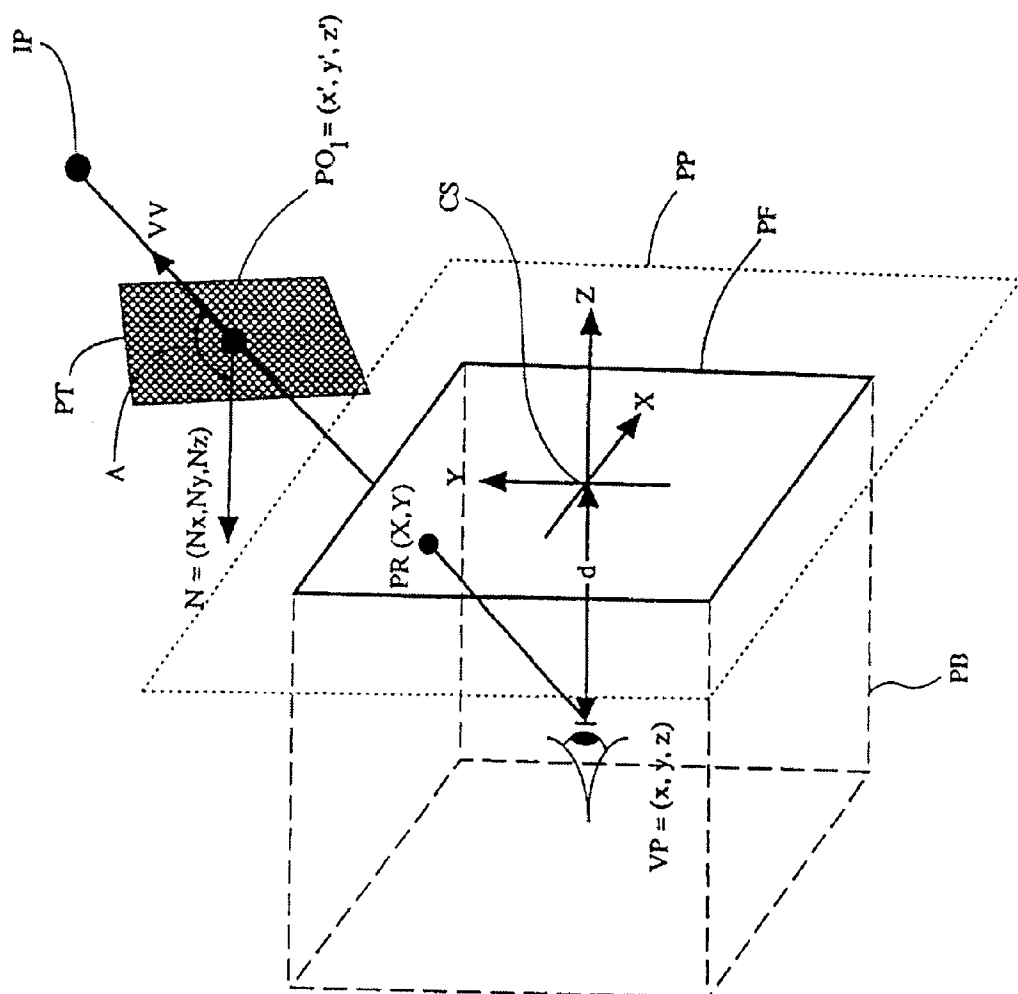


FIG. 2

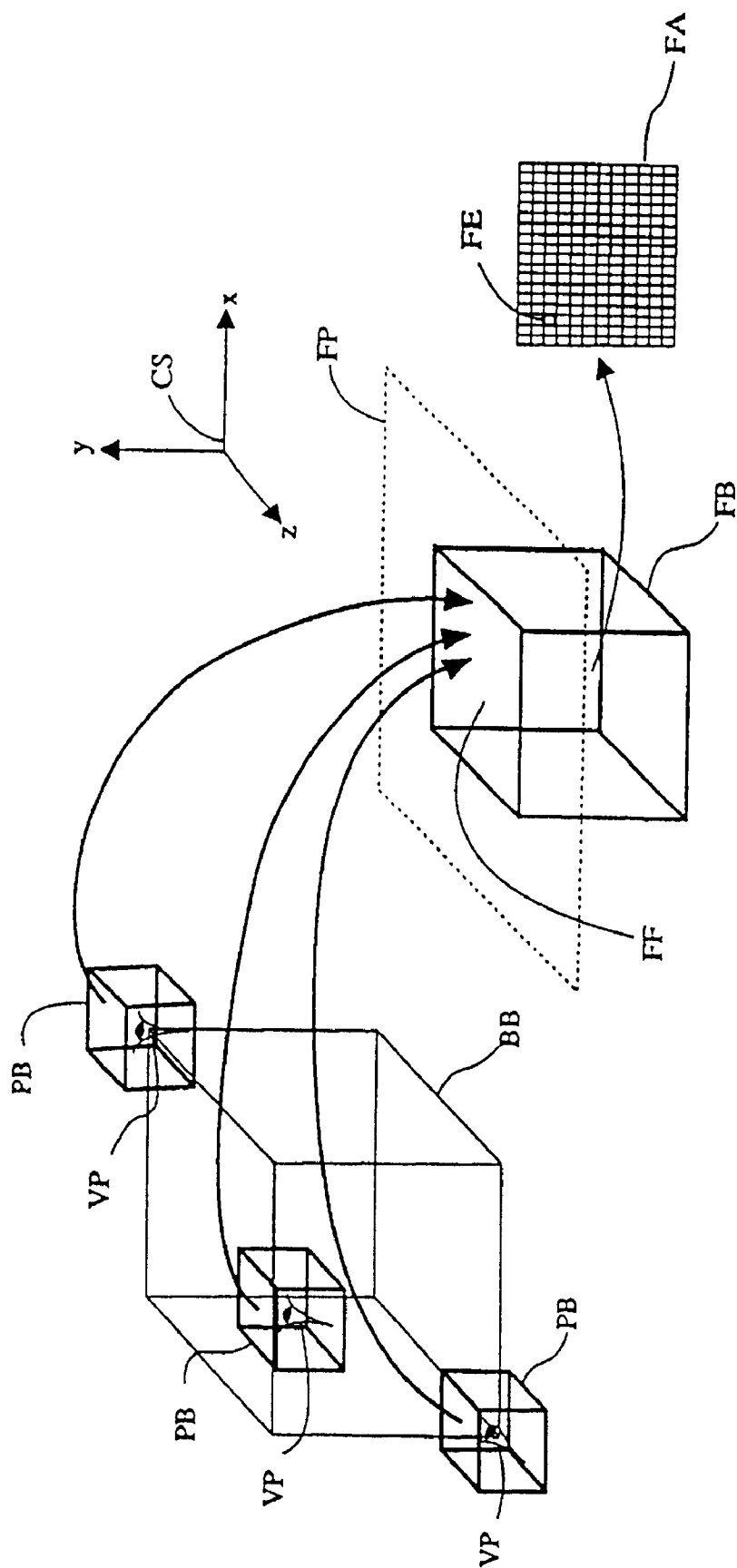


FIG. 3

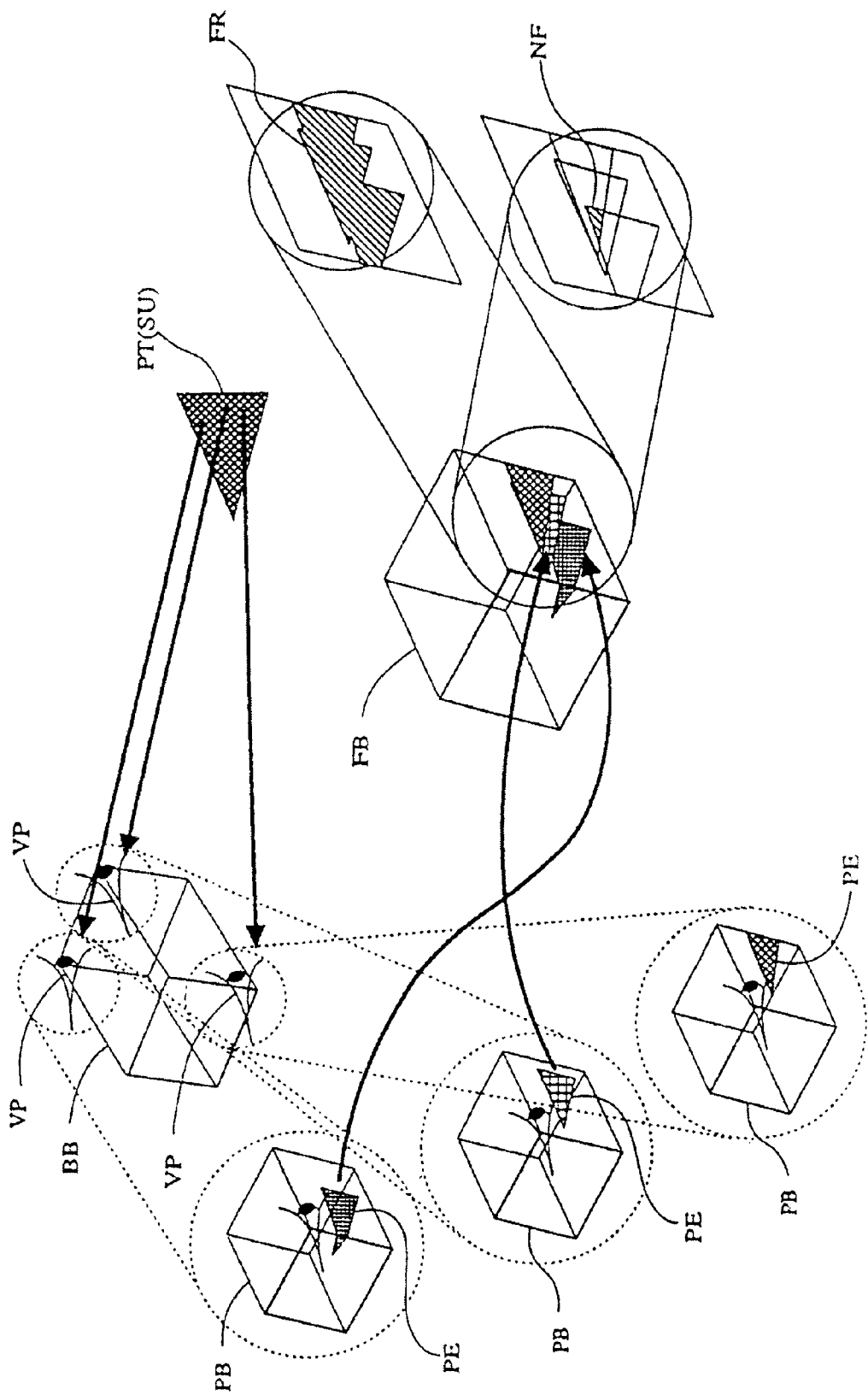
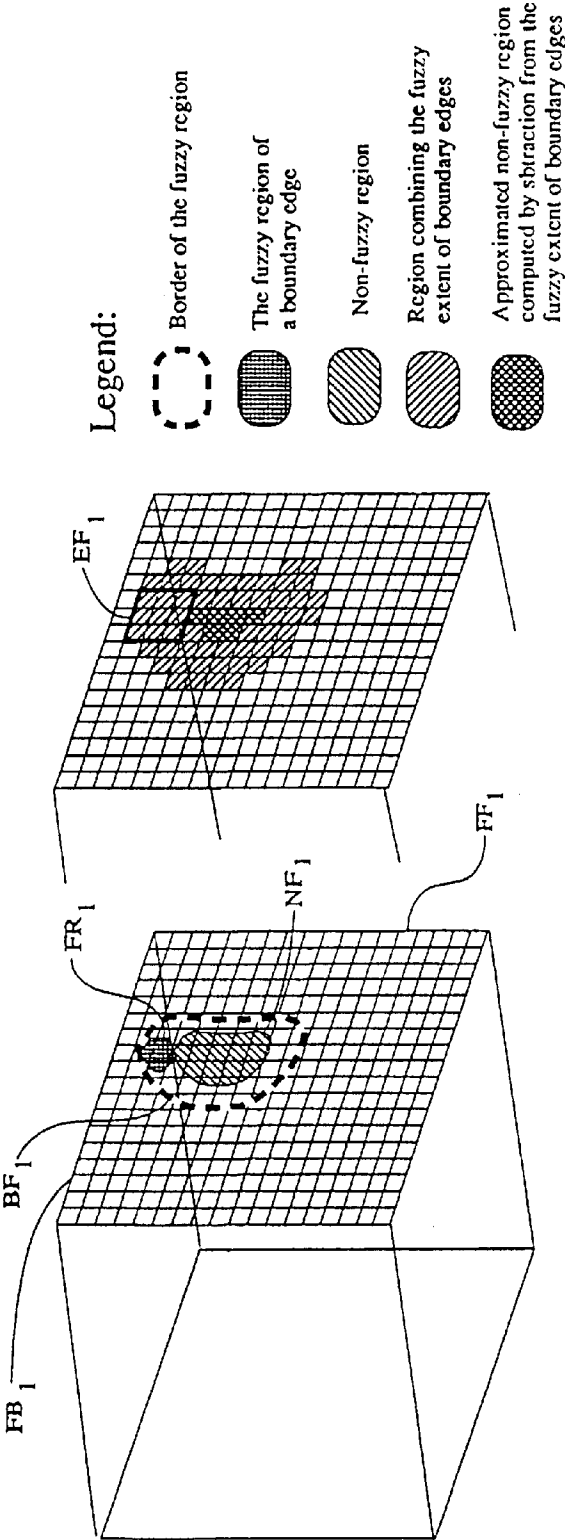
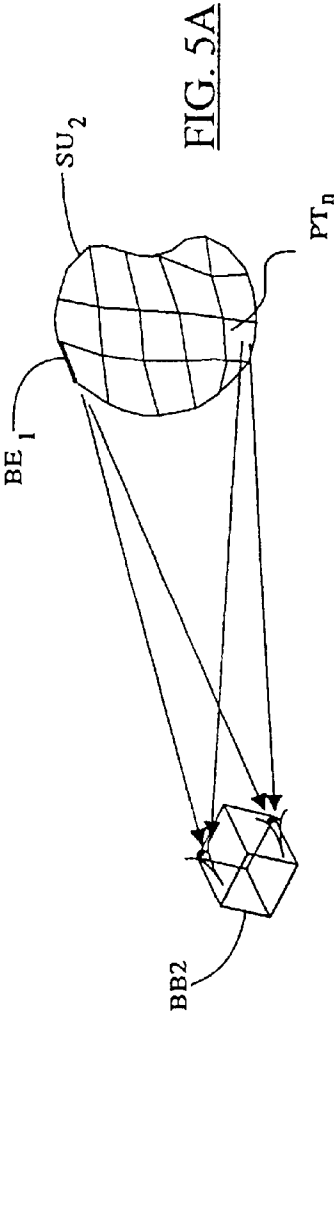


FIG. 4



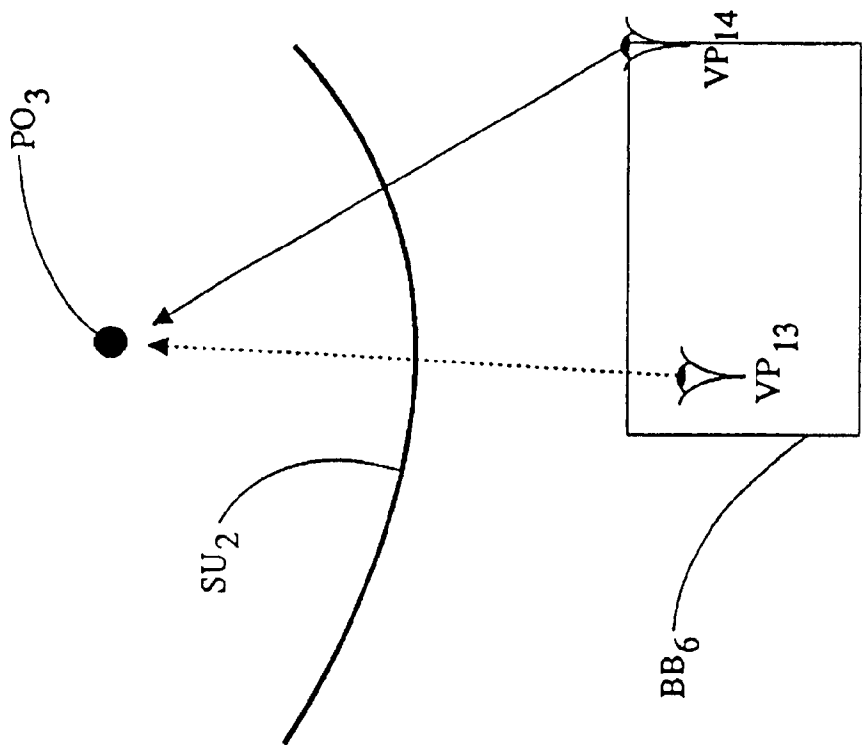


FIG. 5E

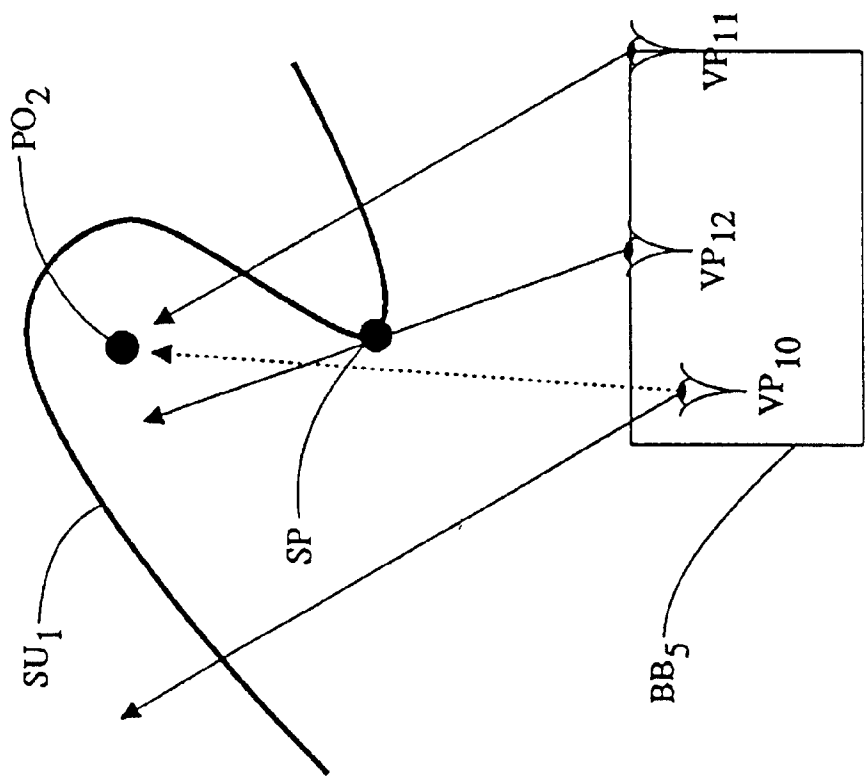
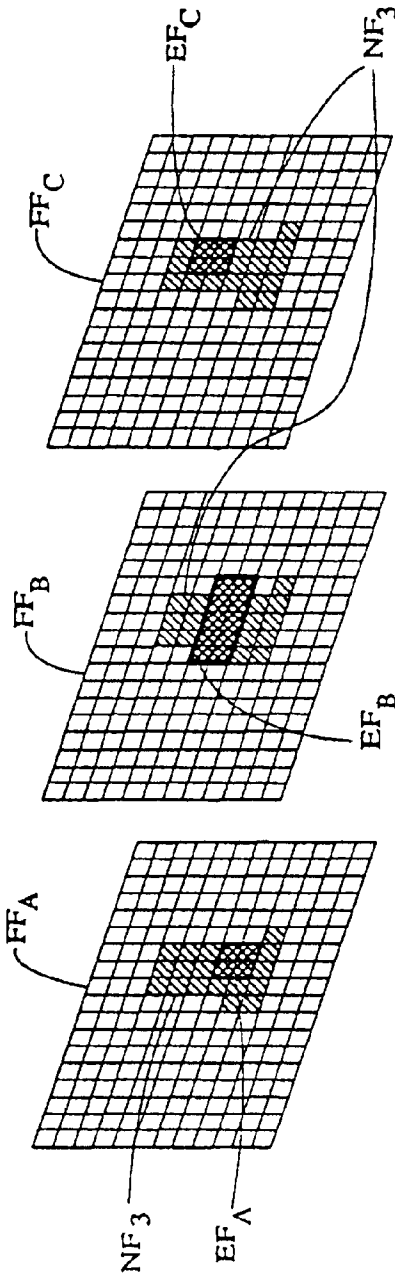
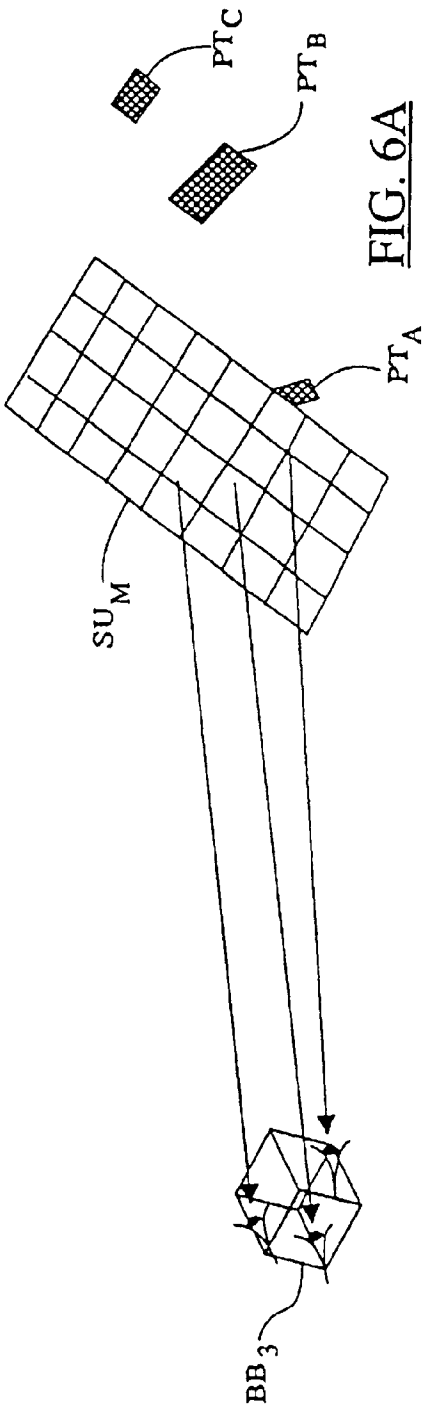


FIG. 5D



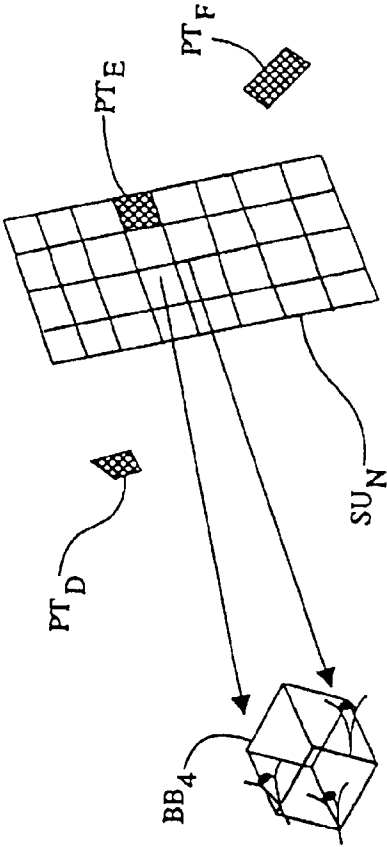


FIG. 7A

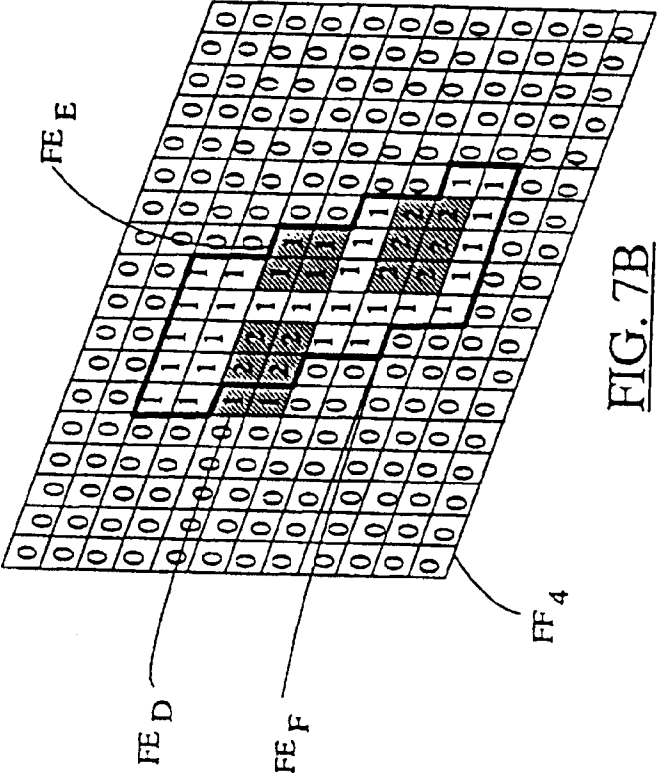
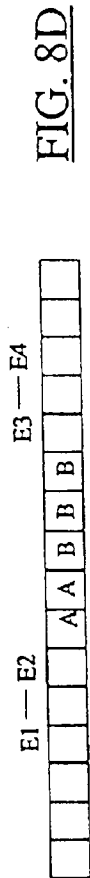
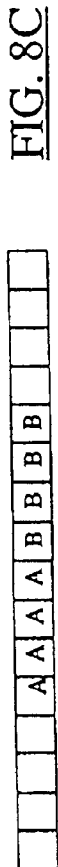
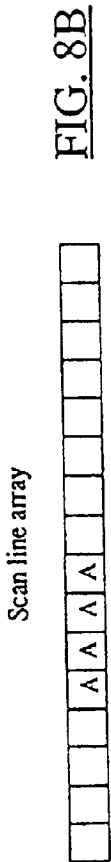
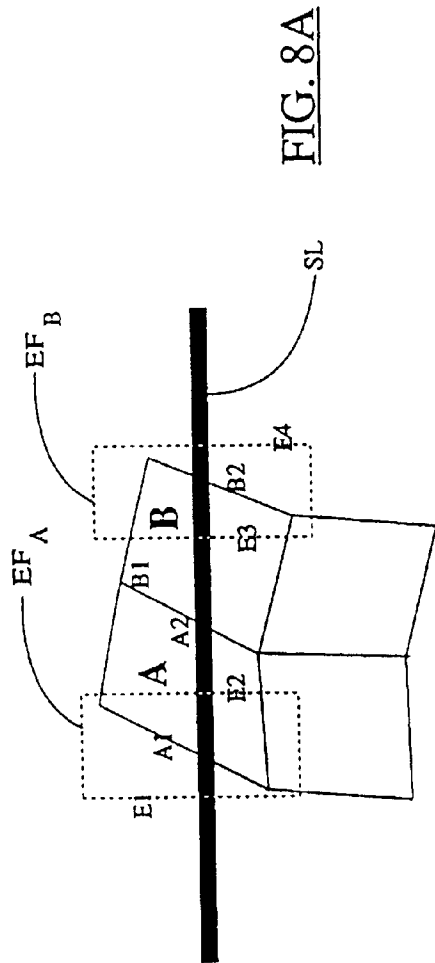
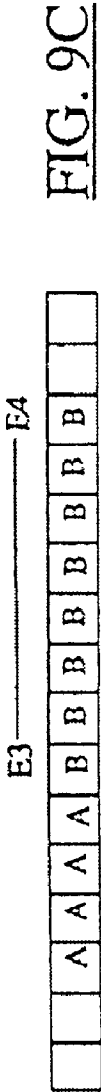
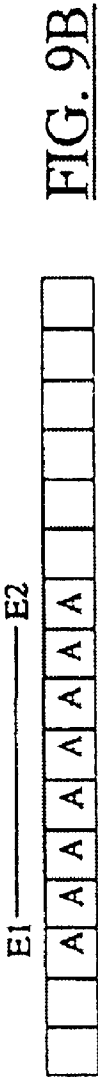
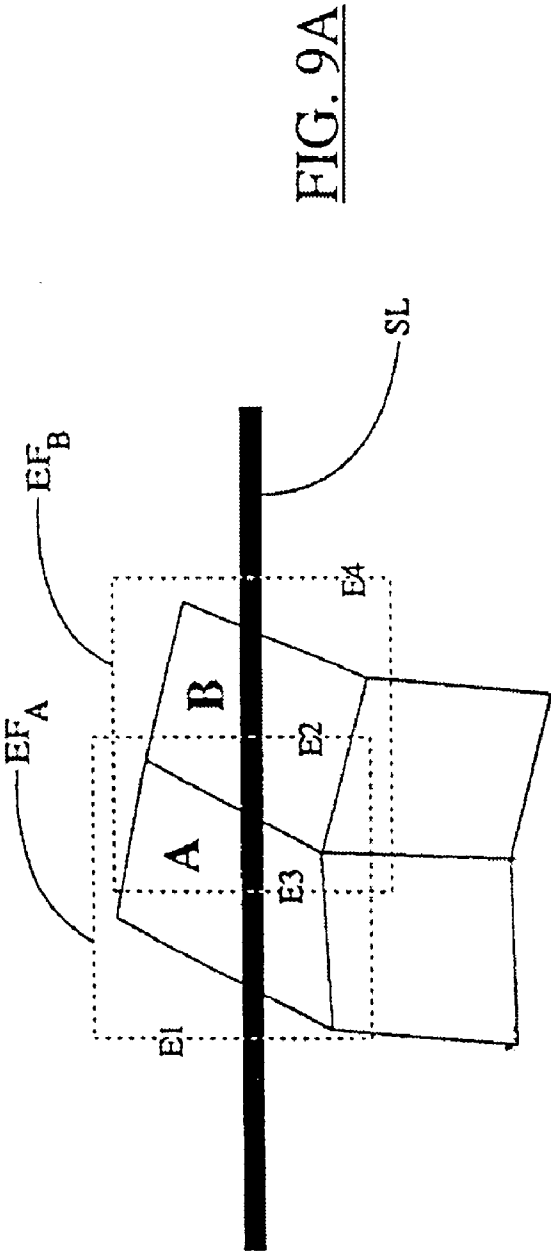


FIG. 7B





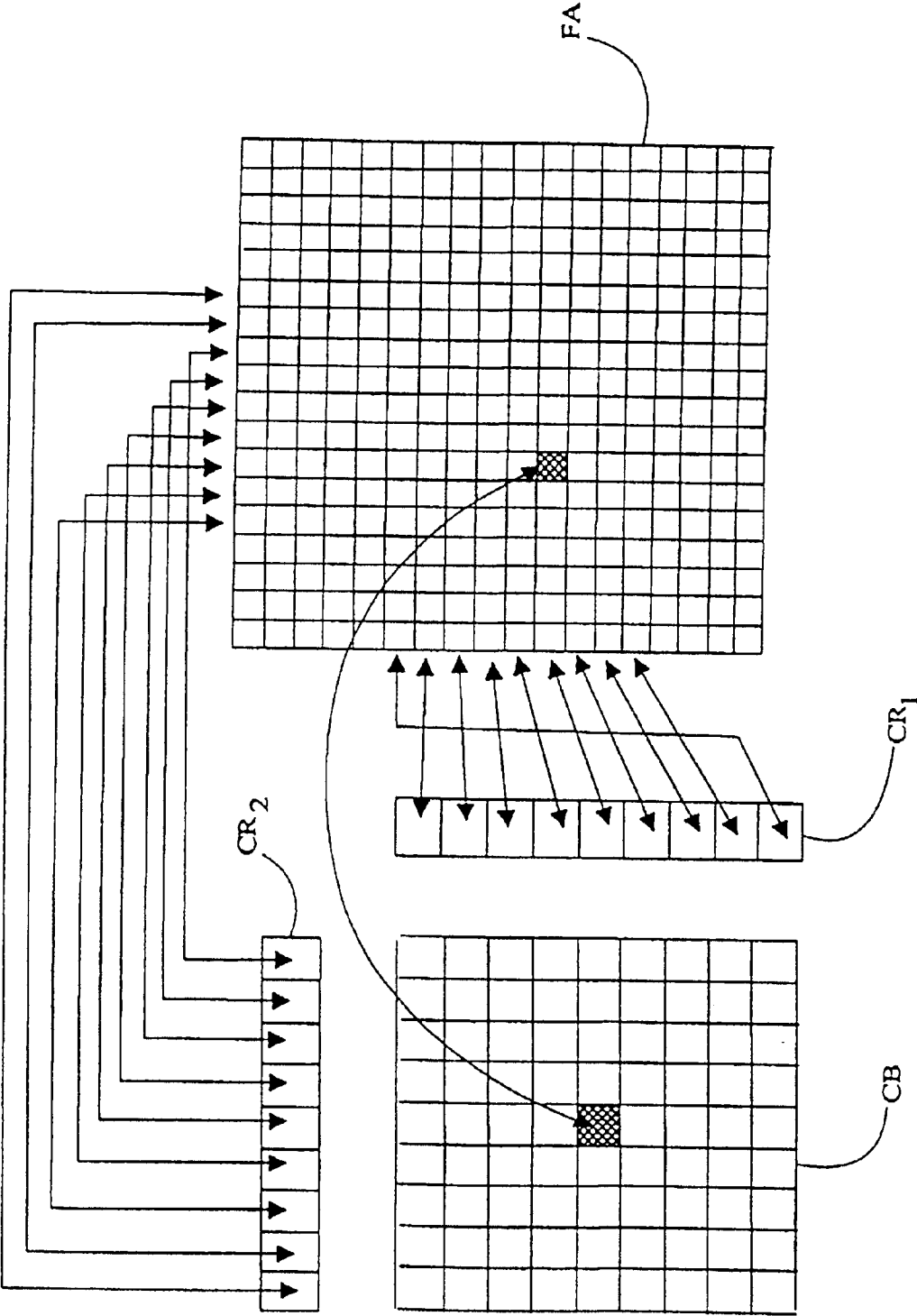


FIG. 10

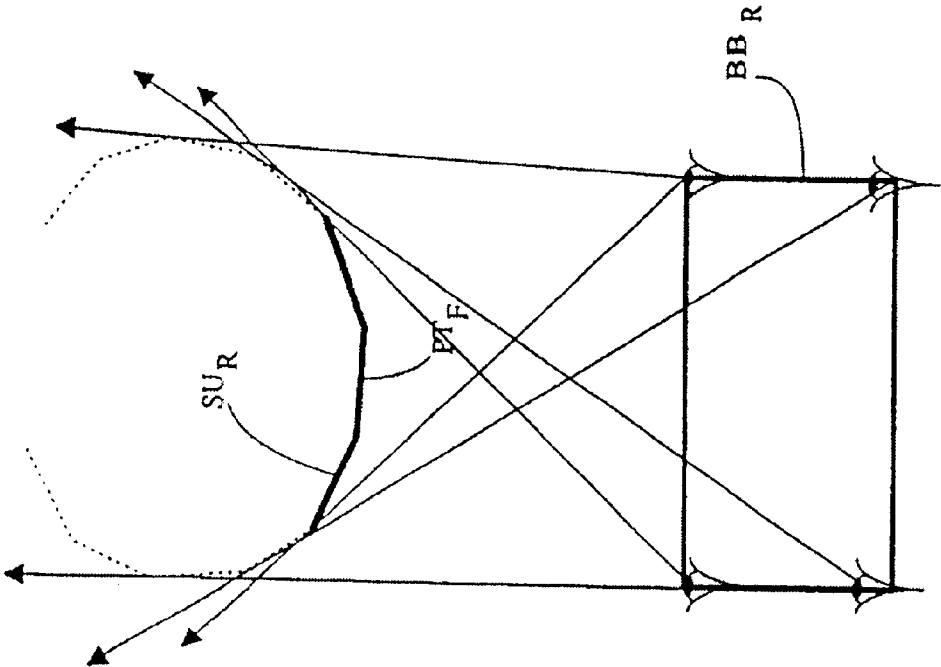


FIG. 11B

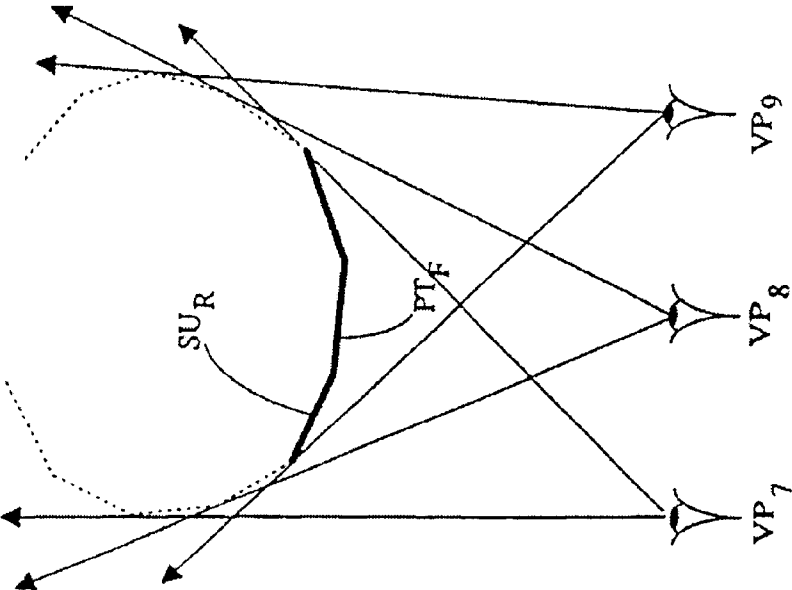


FIG. 11A

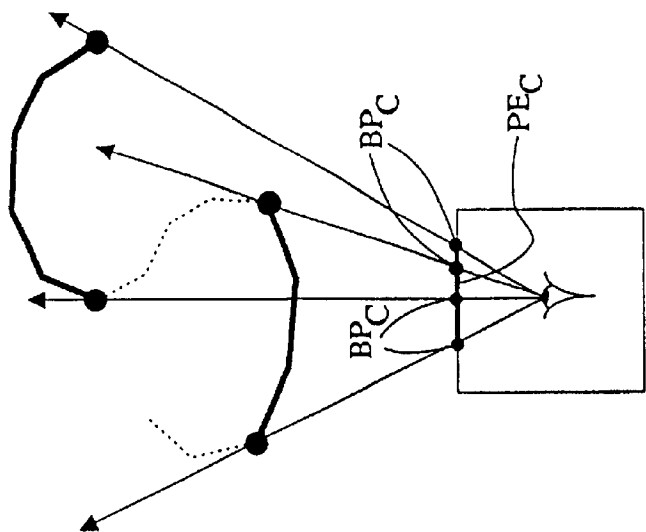


FIG. 12C

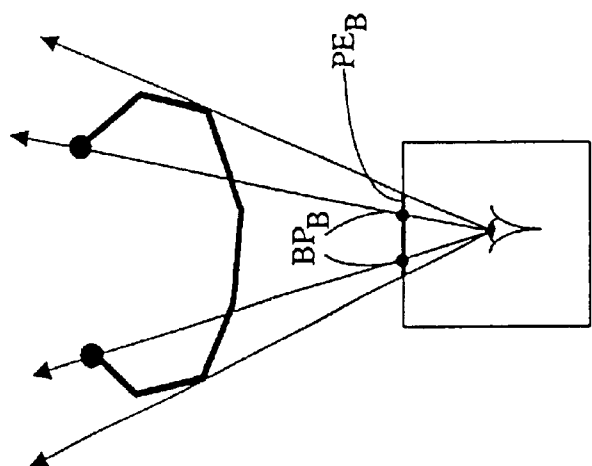


FIG. 12B

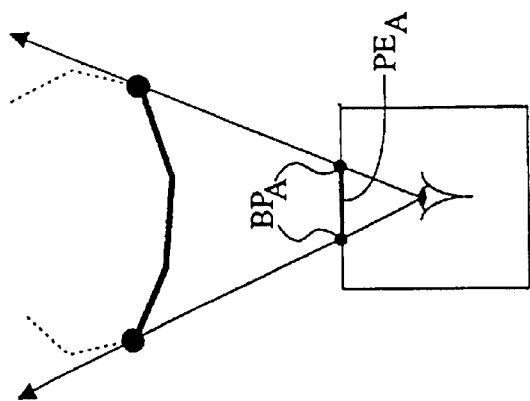


FIG. 12A

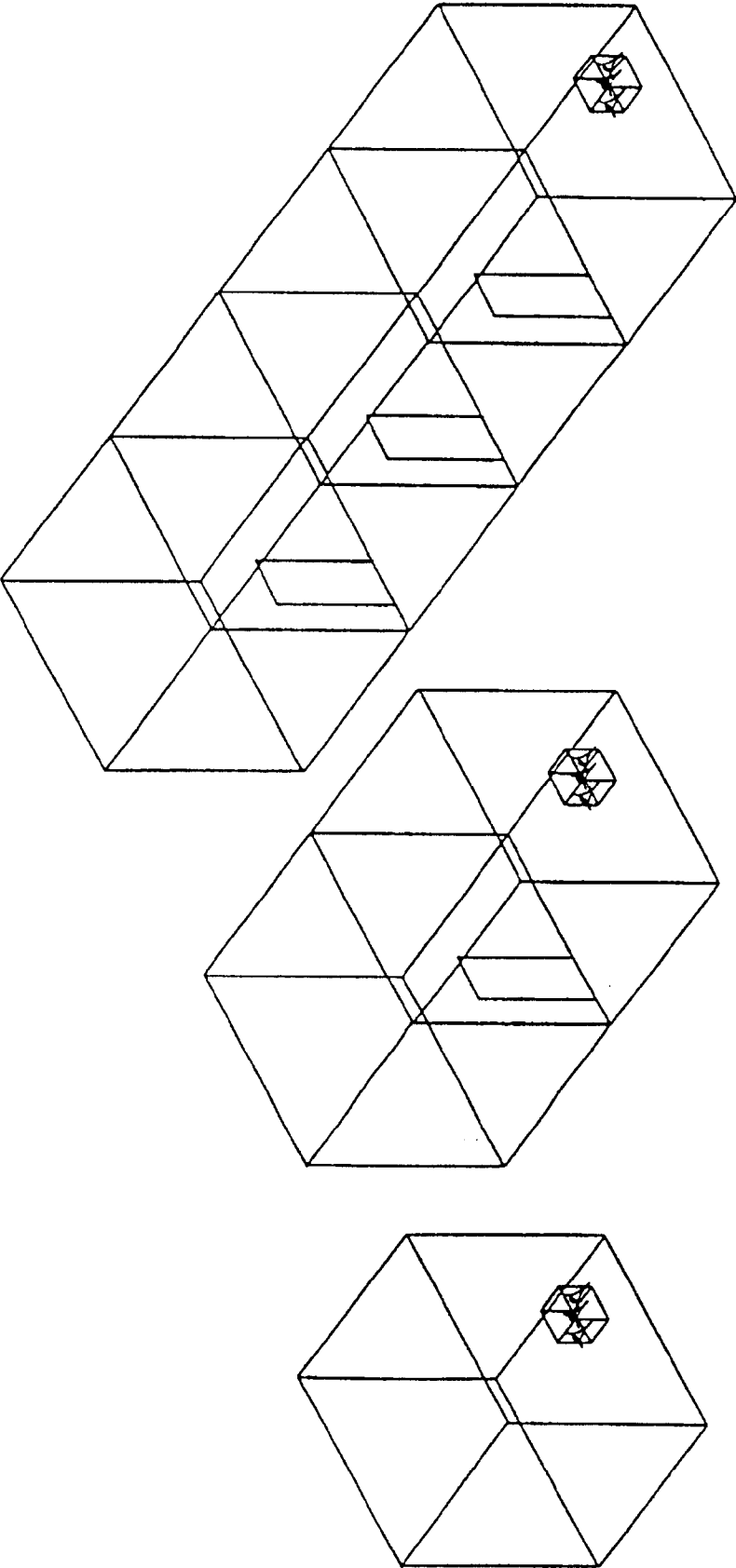


FIG. 13C

FIG. 13B

FIG. 13A

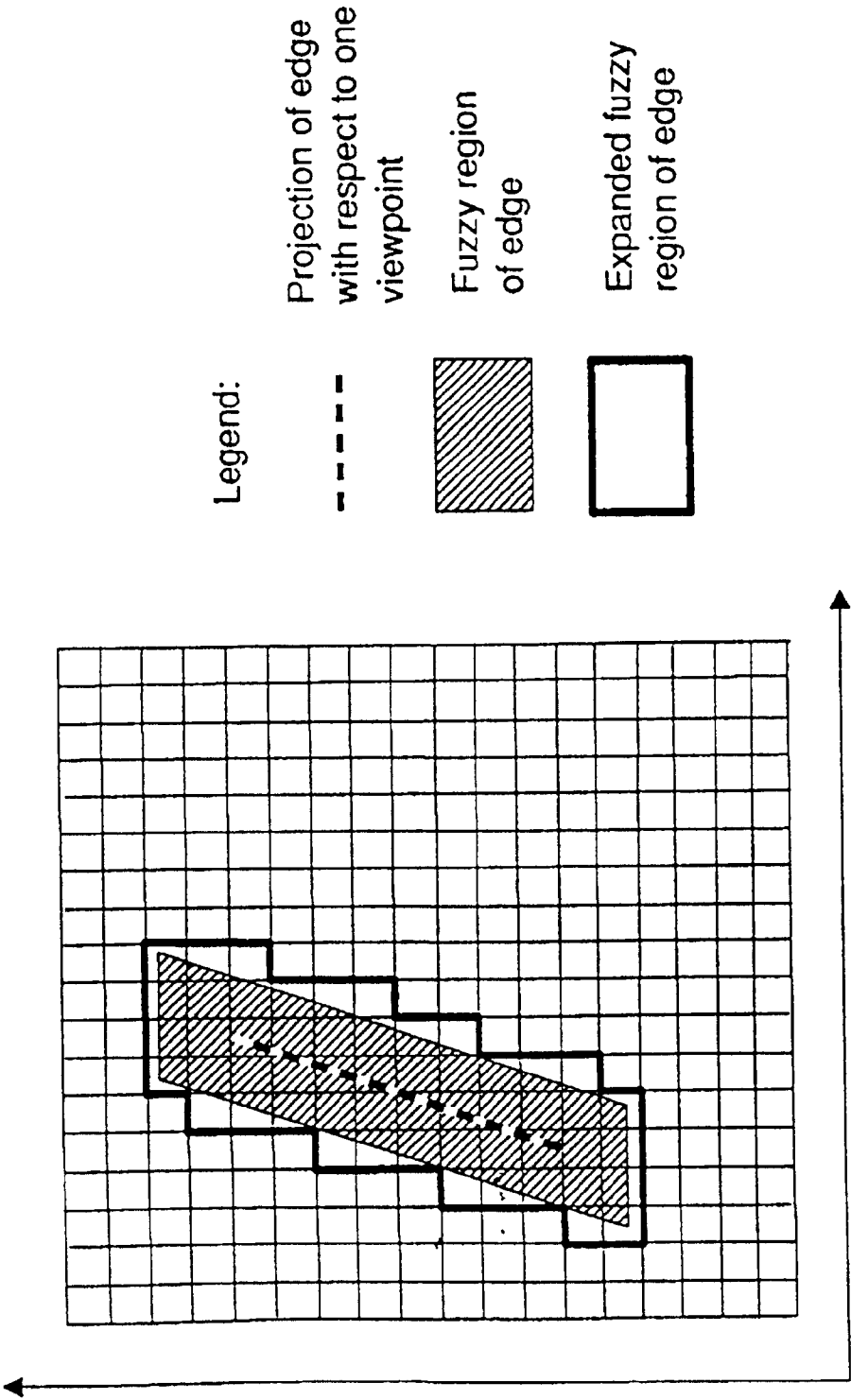


FIG. 14

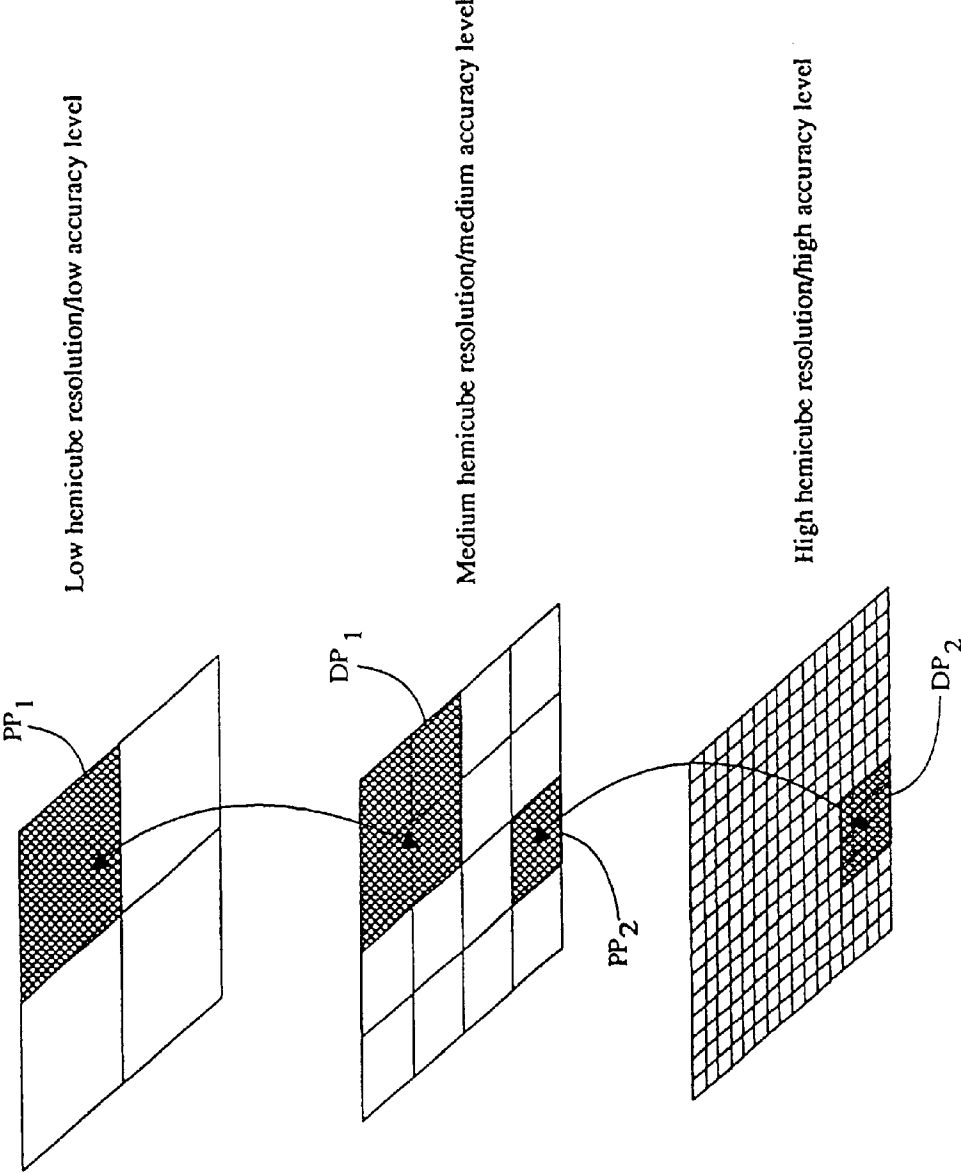


Fig. 15

US 6,618,047 B1

1

VISIBILITY CALCULATIONS FOR 3D COMPUTER GRAPHICS

CROSS RELATED APPLICATIONS

Under 35 U.S.C. §120 this application is a continuation of U.S. Ser. No. 08/935,150, filed Sep. 22, 1997 U.S. Pat. No. 6,172,679, which is a continuation of U.S. Ser. No. 08/182,096, filed Jun. 1, 1994 U.S. Pat. No. 5,914,721, which is a U.S. National Phase filing of PCT application PCT/AU92/00302, filed Jun. 19, 1992.

FIELD OF INVENTION

The present invention relates to computer graphics and, in particular, to the efficient determination of visible and/or invisible surfaces to thereby preferably permit improved hidden surface removal, generally in 3D systems.

BACKGROUND ART

Visible surface detection is one of the most basic operations in 3D graphics. It is applied to generate images of surfaces directly visible to a viewer. Recently, it has also been adopted in the radiosity calculations to compute the energy interactions between surfaces.

The standard strategy of visible surface detection is to divide surfaces into patch elements and compare the spatial relationship between these elements. Using this strategy, the visibility of surfaces cannot be determined until they have been analysed in detail. Although many techniques have been developed to address this issue, none are ideal as they either still require elaborate analysis of surfaces, or they impose various restrictions to the scene.

The limitations of the current techniques can seriously affect the speed of visible surface computation. If the scene is complicated, many surfaces may be invisible. However, the image generation is often slowed down by the need to analyse each surface element in detail. The same limitation has also seriously affected the efficiency of the radiosity computations. Currently, these computations are very slow due to the need to elaborately compute the visibility between every pair of surface elements. Again, these computations may be substantially reduced if surface elements obviously visible or invisible to each other can be more easily computed.

The early visible surface techniques mainly applied various sorting schemes to find the occluding surface primitives. However, with the advancement in hardware technology, it is now common practice to reduce the need for sorting and comparisons by using a large amount of fast memory. This memory may be used to store the object data, such as a BSP-tree. It may also be used to store the depth and surface projection data, as with a z-buffer.

The z-buffer method is simple and has a very low growth rate. However, it still requires depth evaluations and comparisons at each pixel by all the patches projecting onto it, because their visibility is unknown.

The BSP-tree method has the advantage that if the scene is static, an orderly traversal of the BSP-tree is generally sufficient to establish the depth order. However, it still requires the scan-conversion of each path. In addition, the technique needs to re-compute the BSP-tree whenever objects in the scene move. There have been two main strategies of avoiding detailed depth analysis of totally invisible entities. One strategy, applies the property that visibility changes can only occur at the contour edges of surfaces. Visibility computations of internal edges or patches can be reduced by first comparing them with these edges.

2

An alternative strategy is to use the invisible coherence of the scene. These techniques apply the property that an edge is likely to remain invisible in a scan line if it is invisible in the last scan line. Such an edge may therefore be treated specially to avoid unnecessary depth comparisons.

Although the above strategies can reduce some visibility computations, they have limitations. The contour-oriented techniques can operate only in environments which consist exclusively of convex or non-penetrating surfaces. Both the contour-oriented and the scan line approaches are also limited in their ability to reduce visibility computations. In the former, an edge still needs to be tested against the contour edges even if it is invisible. In the latter, all the invisible edge segments at each scan line have to be sorted. These segments also require depth comparisons with the pixels they are on. Finally, all these techniques require some sorting or searching.

SUMMARY OF THE INVENTION

It is an object of the present invention to substantially overcome, or ameliorate, the problems associated with the prior art, through provision of an improved method for performing visibility calculations.

The invention features a method of reducing the visibility related computations in 3-D computer graphics, where the visibility related computations are performed on 3-D surfaces or their sub-elements, or a selected set of both.

In a general aspect of the invention, the method of reducing the visibility related computations includes determining which of the 3-D surfaces or their sub-elements are always invisible or always visible to a viewpoint or a group of viewpoints by projection based computations prior to a visibility computation. The method also includes treating the determined ones of the 3-D surfaces or their sub-elements differently than remaining ones of the 3-D surfaces or their sub-elements during visibility computation.

By determining which of the 3-D surfaces or their sub-elements are always invisible or always visible at the preprocessing stage and by treating the determined ones of the 3-D surfaces or their sub-elements differently than remaining ones during visibility computation, the computation complexity and time request for the visibility computations are significantly reduced.

Embodiments of this aspect of the invention may include one or more of the following features.

The projection based computations include defining projection planes and identifying regions on the projection planes. One or more projection planes are defined with respect to a viewpoint or a group of viewpoints such that projections of selected 3-D surfaces can be formed on them. For example, in one embodiment, identifying regions includes dividing each projection plane into one or more grids and defining a data structure for storing said projections on the grids in computer storage. The grid or grids are either regular or irregular. The data structure of the computer storage is based on a z-buffer or a quadtree.

In another embodiment, the determining step includes identifying the grid cells which are under or related to the projections or extents of projections associated with said 3-D surfaces or their sub-elements and comparing the data associated with said 3-D surface or their sub-elements with the data associated with the grid cells. The data are or related to the depths of said 3-D surfaces or their surface elements.

One preferred embodiment of the treating step is to ignore the determined 3-D surfaces or surface elements which are

US 6,618,047 B1

3

always invisible or always visible during the visibility computation. Intuitively, this scheme results in complexity reduction in the visibility computation. Other features and advantages of the invention will be apparent from the description and drawings, and from the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

A preferred and a number of other embodiments of the present invention will now be described with reference to the drawings in which:

FIGS. 1A and 1B show two viewpoint hierarchies;

FIG. 2 illustrates the projections at an arbitrary viewpoint;

FIG. 3 shows the relationship between projection boxes and a fuzzy projection box;

FIG. 4 shows the creation of fuzzy and non-fuzzy regions;

FIGS. 5A, 5B and 5C show a front-facing surface and its corresponding fuzzy and non-fuzzy regions;

FIG. 5D shows a point that is not totally hidden to a viewpoint bounding box;

FIG. 5E shows a point that is totally hidden to a viewpoint bounding box;

FIGS. 6A to 6D show the detection of totally invisible patches;

FIGS. 7A and 7B show the detection of totally visible/non-hiding patches;

FIGS. 8A to 8D show the scan-conversion of the non-fuzzy regions;

FIGS. 9A to 9C show the scan-conversion of the fuzzy regions; and

FIG. 10 illustrates the mapping between a cache and the fuzzy array.

FIGS. 11A and 11B show views of a front-facing surface region;

FIGS. 12A, 12B and 12C show cut-plane views of the projection of surface regions;

FIGS. 13A, 13B and 13C show respectively one, two and four room models used in testing the preferred embodiment;

FIG. 14 illustrates the direct computation of the fuzzy region of an edge;

FIG. 15 shows a hemi-cube pyramid structure;

Appendix 1 describes The Detection of Patches Front-Facing to a Viewpoint Bounding Box;

Appendix 2 describes The Computation of the Fuzzy Extent of Boundary Edges and Patches;

Appendix 3 describes The Computation of the Non-Fuzzy Regions;

Appendix 4 describes The Scan-Conversion of the Fuzzy Regions;

Appendix 5 describes Using a Cache to Speed up the Access and Writing of Fuzzy Elements Within the Fuzzy Extent of a Patch;

Appendix 6 describes the direct computation of the fuzzy region of an edge;

Appendix 7 describes scan-conversion of patches on the hemi-cube pyramid; and

Appendix 8 is a list of references.

BEST AND OTHER MODES FOR CARRYING OUT THE INVENTION

The preferred embodiments relate to various methods for calculating three dimensional computer graphic images and they are generally termed herein as the "fuzzy projection methods".

4

A Simple Fuzzy Projection Method

In this embodiment a method is provided to map the projections of entities at one or more viewpoints to a set of planes. This embodiment also provides a means to compute regions which contain all these projections and regions which are within each and every projection. The spatial position, time and the optical properties described in this embodiment can be replaced by other physical or abstract variables if the replacement does not affect the mathematical relationship of entities in the graphic image.

Prior to detailed discussion of this and the other embodiments, it is useful to introduce various terms and define variables used in 3D graphics.

Firstly, with reference to FIGS. 1A, 1B and 2, a viewer, as shown by the eyeballs in the FIGS., is an abstract and dimensionless observer where the visibility of its surrounding objects is to be obtained. A point is visible to a viewer if a ray fired from the viewer to it is not blocked by any opaque object. There can be more than one viewer at any time and each viewer may move or rotate over the time. A viewpoint VP is the spatial position and orientation of a viewer at an instance of time. A view vector VV is a vector from the viewpoint VP to a point PO being observed (see FIG. 2).

Several viewpoints VP can be grouped together through certain properties they share. Related groups of viewpoints may in turn form a higher level group. This merging may be repeated to create a viewpoint hierarchy. A current group is the group of viewpoints VP that are currently being considered. In FIG. 1A, viewpoints VP_1 , VP_2 , VP_3 and VP_4 each reside on a path PA_1 and represent viewpoints at times 1, 2, 3 and 4, and can be considered as one group. Similarly, for the viewpoints VP_5 – VP_7 .

Each group of viewpoints is associated with a coordinate system called the group coordinate system CS. A viewpoint bounding box BB of a group of viewpoints VP is the smallest right quadrangular prism enclosing these viewpoints and whose edges are parallel to the axes of the group coordinate system. If the positions or occurrences of a group of viewpoints are not precisely defined, the associated viewpoint bounding box should be the said prism which encloses the space likely to be occupied by these viewpoints. A group of viewpoints may be lying on a plane or a point. In such cases, the viewpoint bounding box may degenerate into a plane or a point. A point PO is totally visible from the bounding box BB if it is always visible from every possible viewpoint VP in the box BB. Conversely, a point is totally invisible from the bounding box if it is hidden from the view of every possible viewpoint VP in the box BB. In FIG. 1A, viewpoints VP_1 – VP_4 reside in a bounding box BB_1 , and viewpoints VP_5 – VP_7 reside in a bounding box BB_2 . The boxes BB_1 and BB_2 can be considered as first level bounding boxes. If the boxes BB_1 and BB_2 are combined, a second level bounding box BB_3 is formed. Rather than the bounding box being a square or rectangular prism, it can also be an ellipsoid, sphere or arbitrary volume.

A viewpoint group may be degenerated and contain only a single viewpoint. In this manner, all the techniques applicable to a normal viewpoint group are also applicable to a single viewpoint. In this case, the viewpoint bounding box BB degenerates into a point which coincides with the viewpoint VP.

In FIG. 1B, multiple viewers at different times are illustrated. A path PA_2 of one viewer of a group is shown extending between locations at two points in time. VP_8 represents the viewpoints at time "1" and VP_9 the view-

US 6,618,047 B1

5

points at time “2”, with BB_2 the bounding box at time “1” and BB_2 the bounding box of the union of the viewpoints VP_8 and VP_9 . Referring now to FIG. 2, shown is a projection box PB, a right quadrangular prism with its centre on a viewpoint VP and its edges parallel to the principal axes of the current group coordinate system CS. The six facets of the box PB are called the projection faces PF. Each projection face PF is on a plane called a projection plane PP. The projection PR of a point PO_1 in the direction perpendicular to a projection plane PP is the intersection between that plane PP and the view vector VV from the viewpoint VP to the point PO_1 . Conversely, since a straight line emanating from the viewpoint VP can only intersect the projection point PR on a projection plane PP, a location on that plane also represents a unique viewing direction from the viewpoint. In this manner:

$$VV=PO_1-VP$$

The area around the visible space point PO, is a visible patch PT and the view vector VV passes through this to an invisible space point IP, occluded by the patch PT.

The projection calculations on each plane are identical to the finding of the image region of an object in the direction perpendicular to that plane. The projection of an object to a projection plane PP therefore represents its image region in the direction parallel to the normal N of that plane.

A viewpoint VP may not be omni-directional but has a limited field of view. Hence certain projection faces may be partially or totally unobservable from the viewpoint. The hidden area of a projection face PF can be obtained by intersecting it with the viewing horizon of the viewpoint VP.

Locations on the projection plane PP can be expressed by a coordinate system CS called the projection coordinates. The origin of this coordinate system is at the centre of the projection face PF. Its x, y axes are parallel to the edges and its z axis is parallel to the normal of the face.

Because they are all parallel, the projection coordinate systems of the current group of viewpoints VP have the same scale factor so that points having the same viewing direction from their viewpoints always have the same projection coordinates.

If points on two surfaces project on the same location, PR for example, on a projection face PF, an occlusion is taking place. The point on the surface further to the viewpoint VP is hidden by the point on the surface closer to the viewpoint VP unless the latter is transparent. This can be determined by the distances d between these points PR and the viewpoint VP.

Every edge of the projection boxes PB of a group is parallel to one of the principal axes (x, y or z) of the respective group coordinate system CS. Since the vectors parallel to the principle axes can have at most six directions, these faces are classified by their normal directions into six sets.

This is shown in FIG. 3 where viewpoints VP_1 , VP_2 and VP_3 , each with respective projection boxes PB_1 , PB_2 and PB_3 , reside within a bounding box BB_1 .

The projections on projection boxes PB_1 – PB_3 can be correlated by another box whose edges are also parallel to the axes of the current group coordinate system CS. This box is called the fuzzy projection box FB. It is called “fuzzy” because it is the combined projection of a single surface (or object, entity, etc.) from all viewpoints, with the combined projection creating a fuzzy image of the surface. Each face of the box is called a fuzzy projection face FF. The face FF lies on a plane called the fuzzy projection plane FP. A projection coordinate system can be defined for a fuzzy projection plane FP and the fuzzy projection face PF on it.

6

Similar to the projection coordinate systems CS of the projection planes, the origin of this coordinate system is at the centre of the fuzzy projection face FF and its axes are respectively parallel to the edges and the normal of the face FF.

By setting the scale factor of the projection coordinate system of each of the fuzzy projection planes to be the same as the scale factor of their associated projection planes PP, each set of projection planes/faces have a one-to-one mapping with the fuzzy projection plane/face that is facing the same direction. Points on a projection plane/face can therefore be linearly mapped to the associated fuzzy projection plane/face.

Points on a set of projection planes having the same projection coordinates, and hence representing the same viewing direction, map to a point on the associated fuzzy projection plane that has the same coordinates. Therefore, similar to the projection planes PP, a point on the fuzzy projection plane FP also represents a unique viewing direction. However, the point is not associated with a particular viewpoint. Since the viewpoints VP can have a limited field of view, some areas of the projection planes may be unobservable. The hidden area of a fuzzy projection face FF is the area where the corresponding areas on all the associated projection faces are hidden. A fuzzy projection face FF is inactive if all its area is hidden.

To sample the projections on the fuzzy projection box FB, each fuzzy projection plane FP is tessellated by two sets of parallel and evenly spaced grid lines forming a fuzzy array FA. By ensuring that there are always grid lines on the edges of the fuzzy projection faces FF, these faces are divided into identical rectangular or square cells. Each cell is called a fuzzy element FE. Each element FE, in addition to representing the viewing direction associated with its centre, also represents a unique solid angle A of viewing directions, as seen in FIG. 2, represented by its perimeter.

Although other representation schemes such as the quadtree subdivision can be applied. All the z-buffer oriented operations would be accordingly changed to operation under such scenes, however the regular subdivision of the faces is considered the most efficient embodiment of the present invention.

Surfaces in the viewing environment are approximated by meshes of patches PT. A surface can also be represented by a hierarchy of patches. Different levels of the hierarchy are meshes of patches at different details. Each patch PT is treated as flat and is approximated by a polygon. The projection of a surface SO to a viewpoint VP is the region combining all the projection regions of its patches. This region is called the projection region PE of the surface. Since the projection region PE of a patch PT represents its image region, the projection region PE of a surface SO also represents the image region of that surface.

The smallest right quadrangular prism just enclosing the patch and whose edges are parallel to the axes of the current group coordinate system is called the bounding volume of the patch. The shortest and longest depths of a patch are the shortest and longest depths between the current viewpoints and points on the patch. The depth corresponds to the z-magnitude in the coordinate direction perpendicular to the projection planes currently being considered. As they are usually difficult to find, they are approximated by the shortest and the longest depths between the viewpoint bounding box and the bounding volume of the patch, which can be determined by partitioning the space by the planes of the viewpoint bounding box and determining which partitions the bounding volume of the patch is in.

US 6,618,047 B1

7

This is shown in FIG. 4 where a patch PT of a surface is viewed from points VP_1 , VP_2 and VP_3 in a bounding box BB_1 . Associated with each viewpoint VP_1 – VP_3 is a projection region PE_1 – PE_3 which is seen to intersect with the corresponding projection face PF_1 – PF_3 in a different manner for each viewpoint VP_1 – VP_3 .

The projections of the patch PT on the same set of projection faces PF_1 – PF_3 can be mapped to the associated fuzzy projection face FF. The areas on the fuzzy projection box FB_1 containing all these mappings are called the fuzzy (combined) regions FR of the patch because logically the region is fuzzy. They cannot tell whether the patch PT1 actually projects on the corresponding area on each projection box PB_1 – PB_3 . Similar to the case of patches, the fuzzy region FR of a surface SU is produced by combining the mappings of the surface projections for a number of patches. In this manner each projection region of the surface is the logical-OR of all the projections of the patches, and the fuzzy projection of each patch is the logical-OR of its projection regions. Accordingly, the fuzzy region FR of a surface SU is also the logical-OR of the fuzzy projections of its patches. This region is equivalent to the superimposition of all the images of the surface seen from the current viewpoints.

On the fuzzy projection box FB_1 , there can also be areas which are always mapped on by the projection regions of the surfaces from the current group of viewpoints. On these areas the logical-AND of the projection regions PE_1 – PE_3 of the surface SU is true. Because the outcomes of the logical-AND operations are always a subset of the outcomes of the logical-OR operations, these areas are always within the fuzzy region FR of the surface SU.

Logically the areas are non-fuzzy as the projection status of the corresponding areas on the current projection boxes is always true. Therefore they are called the non-fuzzy region NF. An alternative term is the umbra region.

The non-fuzzy region NF of a surface SU can be obtained by mapping the projections of the surface at all the current group of viewpoints on the fuzzy projection box FB and finding the area within every projection. However, this is costly if there are many viewpoints. To reduce the computations, a series of approximations which err on the side of caution are applied.

It will be apparent to those skilled in the art that there can exist plural non-fuzzy regions for each surface.

Referring to FIG. 5A, the first approximation assumes that the viewpoints can be at any position in the current viewpoint bounding box BB_2 and each viewpoint has a field of view containing all the possible fields of view of the current group of viewpoints. The patches PT_n facing all these viewpoints are determined. They are called the front-facing patches. A method for finding these patches is described in detail in Appendix 1.

Interconnected front-facing patches PT_n are grouped into surface regions called the front-facing sub-surfaces SU_2 . The edges at the borders of these regions are called the boundary edges BE. Since the front-facing patches PT_n , and their boundary edges BE, are facing all the possible viewpoints VP in the viewpoint bounding box BE, a front-facing sub-surface SU_2 never curves back and is always facing these viewpoints. The projections of its boundary edges BE therefore always surround its projection region PR.

Similar to the patches as shown in FIG. 5B, the projection of each boundary edge BE can be mapped on the fuzzy projection box FB_1 . A fuzzy region FR_1 of the edge BE_1 is the region on the box FB_1 which contains all the projection mappings of that edge at every possible viewpoint in the current viewpoint bounding box BB_2 .

8

The fuzzy regions of all boundary edges belonging to a front-facing sub-surface can be combined into an area called the fuzzy boundary region BF. If a point is not in the fuzzy boundary region BF, the boundary edges BE do not project on all the associated points on the projection boxes. Because any change of projection status must be at the projection of the boundary edges, the point must be within every mapping of the projection regions of the front-facing sub-surface SU, or it must be outside all the mappings.

Since the fuzzy region is the combined projection of the front-facing sub-surface, any area within it must be mapped by at least one projection region of the sub-surface. Therefore, the areas inside it but outside the fuzzy boundary region BF always contain the mappings of the projection regions of the sub-surface. These areas by definition are the non-fuzzy region NF1 of the front-facing sub-surface.

As the viewpoints can have any orientation and be anywhere in the viewpoint bounding box BE, the fuzzy region FR of an edge is difficult to compute. However, the extent of this region on a fuzzy projection face can be more easily obtained. The computation of this extent is described in Appendix 2. The smallest rectangle totally enclosing this extent and with boundary on the grid lines of the fuzzy projection plane is called the fuzzy extent EF of the edge on that plane.

This is seen in FIG. 5C where the same fuzzy projection face of FIG. 5B is shown in which the fuzzy boundary region BF, is replaced by the fuzzy extents EF, of the boundary edges BE_1 . . . etc. The region containing all the fuzzy extents EF of the boundary edges always encloses the fuzzy boundary region of the front-facing sub-surface. Therefore, the subtraction of it from the fuzzy region FR produces an area always within the non-fuzzy region. This area is used to approximate the non-fuzzy region NF.

If a surface contains large patches and hence large boundary edges, the use of the fuzzy extents to approximate its fuzzy boundary region may be inefficient. The fuzzy regions of its boundary edges can be directly evaluated using the method described in appendix 6. Alternatively, each edge can be divided into sub-edges. The fuzzy region of that edge can then be replaced by the fuzzy extents of its sub-edges.

An Invisibility Fuzzy Projection Method for the Detection of Totally Invisible Patches.

In this embodiment a method is provided to compare the mappings of entities on the fuzzy projection faces FF associated with a group of viewpoints VP. Through this operation, entities which may be treated as invisible to all these viewpoints, can be detected.

Referring to FIG. 6A, a viewpoint boundary box BB_3 is shown to observe three patches PA_A , PT_B and PT_C through a mesh surface SUM. FIGS. 6B, 6C and 6D show respectively the fuzzy extents EFA , EF_B and EF_C for the patches PA_A , PT_B and PT_C on respective fuzzy projection faces FF_{A-C} . Also shown is the non-fuzzy region NF_3 of the mesh surface SU_M .

The determination of totally invisible surfaces is based on a property of the non-fuzzy regions. In FIG. 5D, point PO_2 is hidden by surface SU_1 from viewpoint VP_{11} in the current viewpoint bounding box BB_5 . If the point is visible from another viewpoint VP_{10} which is also in BB_5 , then SU_1 has to curve back and reach behind PO_2 . However, in such a situation part of SU_1 next to a point SP would become back-facing to viewpoint VP_{12} in BB_5 . SU_1 would then contain several front-facing surfaces which do not contain the surface around PO_2 . The non-fuzzy region of SU_1 then would not cover the projection mapping of PO_2 . Because of the above property, if the projection mapping of a point from

US 6,618,047 B1

9

an arbitrary point in the Current viewpoint bounding box is within the non-fuzzy region of a surface, and if that point is behind the surface from the viewpoint, that point would always be hidden by the surface from all viewpoints in the box. This is shown in FIG. 5E. Here, the invisibility detection is performed using a single viewpoint (either one of VP_{13} or VP_{14} or other in the box BB_6).

The above observation can be used to detect totally invisible patches, each patch whose visibility is to be determined is selected. The fuzzy elements within the projection mapping of the patch from a viewpoint in the viewpoint bounding box are accessed. If the distance stored in each element is smaller than the depth of the patch from the viewpoint in the direction associated with that element, the patch is an always invisible patch because of the property of non-fuzzy regions just mentioned.

To reduce the above computations, the projection mapping of the patch can be substituted by the extent of that mapping, or the fuzzy extent of the patch. Alternatively, the depth of the patch from the viewpoint may be replaced by the longest possible depth between the patch bounding box and the viewpoint or the viewpoint bounding box. Such approximations always err on the side of caution and do not affect the correctness of hidden surface computations.

A 2D fuzzy arrays FA_A , FA_B , FA_C , with as many storage elements as the fuzzy elements EF on each active fuzzy projection face FF_{A-C} are allocated to store the mapping information on the face. All the fuzzy arrays FAA C are collectively called a fuzzy buffer generally, a memory that is two-dimensionally addressable.

Each fuzzy element EF contains a depth field. The depth field stores the depth of the patch PT whose fuzzy extent EF surrounds the fuzzy element FE.

A cycle of fuzzy buffer computations is applied for each group of viewpoints. First, the depth fields are initialized to infinity to indicate that no surface has yet been projected on the projection box PB.

The fuzzy extents EF of the boundary edges BE and the fuzzy extents EF of the patches PT of a front-facing sub-surface of selected opaque surfaces are computed (as described in Appendix 2). Usually, large surfaces are selected. Based on these extents, the non-fuzzy region NF of the front-facing sub-surface is computed by a scan line technique described in Appendix 3. Furthermore, the fuzzy region of an edge can be found by direct computation as described in Appendix 6. Using the above, the longest depth of the fuzzy elements in each horizontal segment of every non-fuzzy region are obtained. The distance stored in each element of the array is compared with the depth stored in the corresponding element in the fuzzy buffer. If the latter is larger, it is replaced by the former.

After all the non-fuzzy regions of the opaque front-facing sub-surfaces have been updated to the fuzzy elements, each element in the fuzzy buffer contains the longest depth of the closest surface that can be seen from a chosen viewpoint in the current viewpoint bounding box in the direction associated with that element. The fuzzy buffer can then be used to compute patches totally invisible from the current viewpoint bounding box. These patches are deemed to be totally invisible patches. To detect totally invisible patches, the fuzzy elements within the fuzzy extent, the projection mapping, or the extent of the mapping, of each patch are accessed. If the depth stored in each element is smaller than the shortest depth of the patch, the patch is always hidden in the direction represented by that element. Because of the property of non-fuzzy regions, or because of the property that the fuzzy extent covers all the possible viewing direc-

10

tions of the patch, the patch is totally invisible if it is found to be hidden at every fuzzy element within that extent.

As seen in FIG. 6B, the fuzzy extent EF_A of patch PT_A falls entirely within the non-fuzzy region NF_3 . Accordingly, the patch PT_A is hidden from the viewpoints in the bounding box BB_3 .

In FIG. 6C, the fuzzy extent EF_B of patch PT_{VB} both inside and outside the non-fuzzy region NF_3 . Accordingly, those four elements outside may be either visible or hidden, and for the six elements inside, for all view directions through those elements, the patch PT_B is hidden.

In FIG. 6D, the fuzzy extent EF_C of patch PT_C passes the depth test for every fuzzy buffer element it falls on. Patch PT_C is always invisible from the bounding box BB_3 .

Since there is always substantial overlappings between the fuzzy extents of adjacent patches, a high-speed cache can be used to speed up the reading and writing of the fuzzy elements within the fuzzy extent of each patch. The cache can also be used in the visibility fuzzy projection technique discussed in the next section. Appendix 5 describes in more detail the Use of a cache memory.

A Visibility Fuzzy Projection Method for the Detection of Totally visible Patches.

In this embodiment a method is provided to compare the mappings of entities on the fuzzy projection faces associated with a group of viewpoints, Through this operation entities which may be treated as visible and not hiding other entities to all these viewpoints can be detected.

For simplicity and speed, this method assumes that every surface does not hide itself in any viewing direction from the current viewpoint bounding box except those parts which do not face that direction. This implies that no two parts of the same surface that are facing a viewpoint in the bounding box would hide each other. All the planar and quadric surfaces have this property. Other surfaces can be subdivided into smaller surfaces which satisfy this criterion.

As seen in FIG. 7A, a viewpoint bounding box BB_4 observes three patches PT_D , PT_E and PT_F arranged about a mesh surface SUN. Similar to the technique for detecting totally invisible patches, a fuzzy buffer consisting of several arrays is used to store the results of projections on the fuzzy elements.

Every element of the fuzzy buffer contains a field called the homogeneous indicator. This is a three-valued field which indicates whether there is any projection on the associated grid cell on the fuzzy projection box and whether one or more surfaces are being projected onto the cell. The indicator is zero if it is not enclosed by any fuzzy region. The indicator is one if it is inside the fuzzy region of one surface. The indicator is two if it is within the fuzzy regions of more than one surface.

First, the homogeneous indicators are initialized to zero. This shows that no surface has yet been projected on the fuzzy buffer. Using the scan-conversion technique described in Appendix 4, the horizontal segments of the fuzzy region of each surface are obtained. The homogeneous indicators under each segment are examined. Each indicator is incremented by one if it is zero or one.

After the fuzzy regions of all the surfaces have been updated into the fuzzy buffer, each patch is examined. It is ignored if it has been detected as totally invisible by the invisibility fuzzy projection technique.

If the homogeneous indicators within the fuzzy extent of the patch are all one, only one surface can be seen from the current viewpoint bounding box in all viewing directions within that fuzzy extent. The surface has to be the one the patch is on. The patch therefore does not hide other surfaces.

US 6,618,047 B1

11

Since the front-facing parts of the surface do not hide each other, the patch is always visible unless when it is back-facing. The patch is called a totally visible/non-hiding patch. If some of the homogeneous indicators within the fuzzy extent of the patch are not one, the patch might be covering or be occluded by patches of other surfaces at certain viewpoints in the current viewpoint bounding box.

This is shown in FIG. 7B where the fuzzy projection face FF_4 is shown with each element having either the value 0, 1 or 2. As indicated for the patches PT_D and PT_F and their corresponding fuzzy elements FE_D and FE_E , these are not totally visible and not hiding other patches as some of the fuzzy elements within their respective fuzzy extents are mapped to more than one surface. Patch PT_E is totally visible/non-hidden as all its fuzzy elements FEE within its extent are mapped only to one surface (i.e. SU_n).

MODIFIED VISIBILITY FUZZY PROJECTION METHOD

The above technique scan-converts the fuzzy regions of each surface twice because each fuzzy element may be overlapped by the fuzzy regions of several patches of the same surface. This can be avoided by using a linked list called the projected patch list to store the overlapped patches for each fuzzy element. Other data structures such as an array can also be used in place of the linked list. The modified technique consists of the following steps:

1. As the original technique, each fuzzy element contains a homogeneous indicator and a surface ID field. It also contains a projected patch list. The homogeneous indicator and the surface ID fields are initialized to zero and null respectively to indicator that no surface has been projected on the fuzzy element. The projected patch list associated with the element is initialized to the null list.

2. Each patch contains a totally visible indicator. This indicator is initialized to ON. Each patch in the environment is accessed. The fuzzy region of each patch is computed and each element within it is accessed.

3. If the homogeneous indicator of each fuzzy element accessed is zero, the fuzzy element has not been projected onto by the fuzzy regions of any surface. The surface ID of the current patch is written to the surface ID field. The homogeneous indicator is set to one. If the totally visible indicator of the current patch is ON, its ID is added to the projected patch list of the current fuzzy element. Otherwise there is no such necessity because the projected patch list is used to reset the totally visible indicators of patches.

4. If the homogeneous indicator is one, the surface ID of the current patch is compared with the surface ID stored in the surface ID field of the current fuzzy element. If they are the same and the totally visible indicator of the patch is ON, the ID of that patch is added to the projected patch list associated with the current fuzzy element. If the ID's of the two surfaces are different, the homogeneous indicator is set to two to indicate that there is more than one surface projecting onto the current fuzzy element. The current patch and all the patches with ID's stored in the projected patch list of the current element may not be totally visible. Therefore, their totally visible indicators are set to OFF. Records in the projected patch list can be removed after all their totally visible indicators have been updated.

5. If the homogeneous indicator is two, the current fuzzy element has already been found to be projected onto by more than one surface. The current patch therefore may not be totally visible. The totally visible indicator of its record is set to OFF.

12

6. After the processing of all patches a patch is determined to be totally visible if its totally visible indicator has not been set to OFF.

The pseudo-code of the above processing is shown below:

```

initialize the totally invisible indicators of all patches to ON;
initialize the homogeneous indicator of all fuzzy elements to zero;
clear all projected patch lists of the fuzzy elements;
for (each patch of the current surface)
do
  for (each fuzzy element under the fuzzy region of the patch)
  do
    if (the homogeneous indicator is zero)
    then
      write the surface ID of the current patch to the surface ID field;
      set the homogeneous indicator to one;
      if (the totally invisible indicator of the current patch is not OFF)
      add the patch ID to the projected patch list of the fuzzy element;
    endif
    else if (the homogeneous indicator is one)
    then
      if (the surface ID of the current patch is the same as the surface ID stored in the fuzzy element)
      then
        if (the totally invisible indicator of the current patch is not OFF)
        add the patch ID to the projected patch list of the fuzzy element;
      endif
    else
      set homogeneous indicator to two;
      set the totally invisible indicator of the current patch and each patch indexed by the projected patch list of the fuzzy element to OFF;
    endif
  else
    set the totally invisible indicator of the current patch to OFF;
  endif
done
done

```

Use of the Visibility and Invisibility Methods for the Radiosity Methods By the application of the foregoing methods, a list of the totally invisible patches and the totally visible/non-hiding patches can be obtained for a viewpoint group. The visibility of these patches remains the same for all the subgroups and viewpoints under that group. Therefore these patches need not go through elaborate visibility computations in these subgroups and viewpoints.

For each viewpoint in the group, the totally invisible patches may be ignored. The totally visible/non-hiding patches need not be compared with other patches to determine their visibility and form-factors. Their form-factors can be computed directly from their orientations, positions and shapes using mathematical formulas describing the radiative transfer between surfaces.

US 6,618,047 B1

13

If a patch is bright or close to the current group of viewpoints, the direct form-factor computations need to be carried out in higher accuracy. The patch can be subdivided into smaller patches and the form-factor computations are carried out for these sub-patches instead. Alternatively, accurate radiative transfer formulas such as the Nussel analog technique can be used. This technique finds the fractional area of the patch after it has been projected onto the surface and then the base of a hemisphere whose centre is at the viewpoint.

If accurate form-factor computations are not required, the form-factor of a patch for the current group of viewpoints may be obtained by assuming that it is constant throughout that patch. The standard form-factor equation can be simplified to:

$$F = \cos A \cdot \cos B \cdot A / (\pi r^2)$$

where A and B respectively are the angles between the view vectors and the normals of the observed patch, and the patch the viewpoint is on, and r is the distance between the viewpoint and the centre of the observed patch.

The fuzzy projection methods require orderly top-down traversal of viewpoints according to their hierarchy. However, in techniques such as the progressive refinement methods, the receiving patches are usually accessed in decreasing order of their brightness. Therefore, if these techniques are used in conjunction with the fuzzy projection methods, the form-factor computations of some viewpoints in a group may not have been carried out since not all the viewpoints within the group may be processed in one go. Information about which are the totally visible/non-hiding patches and which are the totally invisible patches of the partially completed group has to be stored in memory. The outstanding viewpoints in the group can then use this information when their turn to be processed has arrived. This information can be removed when all the viewpoints in the group have been processed.

In techniques such as the progressive refinement methods (known per se), the form-factors between patches have to be repeatedly computed. Therefore, if memory permits, the information about which patches are totally visible/non-hiding and which patches are not totally invisible may not be removed and can be advantageously used.

In the first round of computations, all the fuzzy projection and normal hemicube computations for every receiving patch are carried out as required. During these computations, the emission patches at each group/viewpoint are classified into three groups: totally visible/non-hiding, totally invisible, and the remainder. This classification information is stored using data structures such as arrays or linked lists.

In subsequent rounds of computations, where the form factors between patches have to be re-evaluated, the fuzzy projection computations are not repeated. The totally visible/non-hiding and the totally invisible patches of each viewpoint can be simply retrieved from the stored information.

The memory for storing the classification of patches for each group of viewpoints can be reduced by several methods. First, since patches found to be totally visible/non-hiding or totally invisible from a group will maintain the same status for all its subgroups, they need not be repeatedly stored.

Secondly, because the union of the three lists of patches is the set of all the patches that need to be classified for each group of viewpoints, only two lists need to be stored for the group. To further reduce storage, the smallest pair of the three lists can be kept. Also, because the records of patches

14

are usually orderly organized and accessed, the patch IDs in the lists usually have some order. The differences between the IDs of successive patches in the lists are often much smaller than the magnitude of these IDs. Whenever this occurs, these differences instead of the patch IDs can be stored. A sign bit can be used to distinguish between the ID's and difference values. The list can be sorted by the ID for more efficient use of this arrangement. Finally, the visibility status of patches often do not change much from one group of viewpoints to an adjacent group of viewpoints. Therefore, using the patch lists of a viewpoint group as a starting point, a series of nearby groups can store their patch lists incrementally. Starting from the first group, each adjacent group only needs to store the lists of patches that need to be deleted from, and added to the patch lists of the previous group to form the current lists. Again, a special field can be used to distinguish the use of this arrangement.

ENHANCING THE ACCURACY OF FORM-FACTOR COMPUTATIONS

The fuzzy projection techniques can also be used to achieve optimal accuracy of the form-factor computations. After the filtering off of totally invisible patches by the invisibility technique, the visibility technique detects totally visible and non-hiding patches. As mentioned above, different methods which yield different accuracy in the computations of the form-factors of these patches can be applied. The remaining patches are not totally invisible. They are also not totally visible/non-hiding. The accuracy of their form-factor computations can be determined by the following steps:

a. First, several levels are defined such that each of them corresponds to a level of accuracy to be reached by the form-factor computations of patches. Each accuracy level determines the strategies of form-factor computations such as the resolution of the hemicube buffer, the level of details of patches, whether to use ray-tracing instead of the hemicube buffer, or the number of rays traced per sampling point.

b. All the patches not found to be totally invisible or totally visible are classified by their accuracy level according to their brightness, importance, distance from the current group of viewpoints, and other properties. Each patch record has a field which stores its accuracy level. An accuracy level field is also allocated for each fuzzy element in the fuzzy buffer. Each field is initialized to the lowest accuracy level.

c. Patches are accessed in the order such that those having higher accuracy levels are scan-converted first.

For each patch accessed, the same scan-conversion of their fuzzy extents applied in the visibility fuzzy projection computations is carried out. However, instead of the homogeneous indicators, the accuracy level fields in each fuzzy element access is examined. If the value in a field is larger than the accuracy level of the patch, the patch is likely to be hiding or be hidden by a patch at a higher accuracy level. The accuracy level of the patch is set to this value. If the value is smaller than the original accuracy level of the patch, the later is written to the former.

d. After the processing of all patches, the updated accuracy level of each patch shows the maximum level of accuracy its form-factor computations needs to be carried out for the current group of viewpoints. If this level is higher than the accuracy the patch can provide, it may be recursively subdivided and replaced by sub-patches whose accuracy matches the level.

If the actual surface of a patch is curved, the projections of its sub-patches may be outside its projection. Therefore,

US 6,618,047 B1

15

for the subdivision strategy to be effective, the fuzzy extent of each entity should contain the fuzzy extents of all its sub-entities during all the fuzzy projections. It can be approximated by the fuzzy extent of the axis-aligned bounding box containing all these sub-entities.

e. The accuracy level field of a fuzzy element indicates the accuracy of the computations that needs to be carried out in the corresponding viewing directions from the current group of viewpoints or the corresponding regions on the hemispheres of these viewpoints. The form-factor of a patch can be computed in variable resolutions which match the different accuracy requirements of the fuzzy elements covered by that patch. This can be done in either the ray-tracing or the hemisphere approach of form-factor computations.

f. In the ray-tracing approach, the form-factors of patches at a viewpoint is computed by tracing rays in all observable directions. For each of the current group of viewpoints, the number of rays to be fired within a solid angle associated with a fuzzy element depends on the accuracy level of that element. If the level is high, more than one ray should be traced. Conversely, if the accuracy levels of several adjacent fuzzy elements are low, only one ray may be traced for the directions associated with these elements.

9. In the hemisphere approach, different regions on the hemisphere should have different sampling resolutions which match the accuracy requirements of the fuzzy elements associated with these regions. This can be achieved by the use of a pyramidal representation called the hemisphere pyramid. The scan-conversion of patches on the hemisphere pyramid is described in Appendix 7.

Use of the Visibility and Invisibility Methods for Hidden Surface Computations

In this embodiment a method is provided to apply the computed results of the visibility and invisibility fuzzy projection methods in the hidden surface computations.

In shadow computations, in applications where a series of images are to be taken, such as computer animation, flight simulation, or dynamic graphics, the hidden surface removal operations need to be repeated many times. The position of the viewer when an image of the surrounding is taken may be treated as a viewpoint. The viewpoints may also be combined into a hierarchy of viewpoint groups.

If the combined field of view of the current viewpoint group is narrow, the projection boxes and the fuzzy projection box can be flattened so that only a fuzzy projection plane and its associated projection planes are active for the current group of viewpoints. By using the invisibility fuzzy projection method, patches found to be totally invisible to a group of viewpoints, do not need to be considered for the subgroups and the individual viewpoints in the group. By using the visibility fuzzy projection method no depth computations and comparisons of totally visible patches during the scan-conversion is necessary.

Use of the Visibility and Invisibility Methods for the Ray Tracing Computations

In this embodiment a method is provided to apply the computed results of the visibility and invisibility fuzzy projection techniques in the ray tracing techniques.

In a ray tracing application, the objects and surfaces in the environment are often organized as a hierarchical geometric model. If the number of rays traced from a node of the

16

geometric model justifies the use of the fuzzy buffer methods, a viewpoint group is created by treating every point on the surfaces of that node where a ray might emit as a viewer. The position of the viewer at any instance of time is a viewpoint. The viewpoints can be grouped into a hierarchy of viewpoint groups.

From the fuzzy buffer computations, patches which are totally visible/non-hiding to a group of viewpoints corresponding to each node and patches which are totally invisible to this group of viewpoints. When a ray is traced from a node, all the totally invisible patches need not be considered. Also, the ray is tested first with the totally visible/non-hiding patches. If the ray hits one of those patches, no further testing with other patches need to be carried out as the patch being hit is the closest one on the path of the ray.

If both the radiosity method and the ray tracing techniques are used, the results of the fuzzy buffer computations can be used for both methods.

USE OF THE VISIBILITY AND INVISIBILITY FUZZY PROJECTION TECHNIQUES IN COMPUTER VISION

The visibility and invisibility fuzzy projection techniques can be used in computer vision. The occlusion analysis of objects is a vital operation in computer vision. Based on this analysis, hypotheses are made to construct a 3D model. This model is then matched with the vision data or existing models. It may be repeated matched and refined until it is acceptable. Usually there are a lot of uncertainty and vagueness in the vision data and the hypothesis. Such imprecision can be accommodated by the present techniques in the following ways:

1. During the model construction phases, a viewpoint bounding box can be defined which include all the likely positions of the viewpoint.

2. During the model construction phase, a 3D model is generated based on the vision data. However, if the exact locations and shapes of entities such as edges and patches are uncertain, the entities may be approximated by the bounding boxes which contain all their likely locations.

3. Based on the hypothesized model, the invisibility and visibility techniques are carried out to compute totally visible surfaces, totally invisible surfaces and surfaces whose visibility cannot be determined.

4. The totally visible surfaces correspond to the vision data whose information is relatively explicit. Such data may be treated as successfully interpreted or more cursorily checked. Areas on the fuzzy projection box which are projected onto by surfaces whose visibility cannot be determined correspond to the vision data which are more obscured. The areas on the image plane corresponding to these areas are further analyses.

5. In the next round of analysis, the hypothesis becomes more refined. The viewpoint bounding box and bounding boxes of entities may be accordingly shrunken.

The preferred embodiment has been implemented on a general purpose computer adapted via programming in the C-language and tested using a set of models which differ in depth complexity. Three of the models are shown in FIGS. 13A, 13B and 13C. The visible surface computations have been computed for five nearby viewpoints. The times of hidden surface computations under different pixel and patch resolutions have been measured and are shown below.

US 6,618,047 B1

17
EXAMPLE 1

Coarse patches, coarse screen pixels (400×400 pixels), resolution of fuzzy buffer: 400×400 pixels

Model	Normal Hidden Surface Removal		Fuzzy Hidden Surface Removal		Patches invisible
	CPU time for 5 viewpoints (sec)	Total Patches	CPU time for fuzzy computations(sec)	CPU time for 5 viewpoints (sec)	
1 room	11.2	10807	1.2	11.1	157
2 room	21.8	19137	2.1	17.3	3942
3 room	35.7	27467	2.9	23.9	9041
4 room	46.3	35797	3.6	25.6	16140

EXAMPLE 2

Coarse patches, fine screen pixels (1000×1000 pixels), resolution of fuzzy buffer: 400×400 pixels

Model	Normal Hidden Surface Removal		Fuzzy Hidden Surface Removal		Patches invisible
	CPU time for 5 viewpoints (sec)	Total Patches	CPU time for fuzzy computations(sec)	CPU time for 5 viewpoints (sec)	
1 room	20.7	10807	1.2	20.7	157
2 room	40.8	19137	2.1	32.4	3942
3 room	56.4	27467	2.9	37.8	9041
4 room	72.9	35797	3.6	40.0	16140

EXAMPLE 3

Fine patches, coarse screen pixels (400×400 pixels), resolution of fuzzy buffer: 400×400 pixels

Model	Normal Hidden Surface Removal		Fuzzy Hidden Surface Removal		Patches invisible
	CPU time for 5 viewpoints (sec)	Total Patches	CPU time for fuzzy computations(sec)	CPU time for 5 viewpoints (sec)	
1 room	25.0	21263	2.0	24.1	1605
2 room	41.1	38293	2.5	25.0	15301
3 room	60.6	55323	3.4	29.1	28712
4 room	80.7	72353	4.6	33.8	42103

EXAMPLE 4

Fine patches, fine screen pixels (1000×1000 pixels), resolution of fuzzy buffer: 400×400 pixels

Model	Normal Hidden Surface Removal		Fuzzy Hidden Surface Removal		Patches invisible
	CPU time for 5 viewpoints (sec)	Total Patches	CPU time for fuzzy computations(sec)	CPU time for 5 viewpoints (sec)	
1 room	42.0	21263	2.0	40.5	1605
2 room	75.9	38293	2.5	45.4	15301

18

-continued

Model	Normal Hidden Surface Removal		Fuzzy Hidden Surface Removal		Patches invisible
	CPU time for 5 viewpoints (sec)	Total Patches	CPU time for fuzzy computations(sec)	CPU time for 5 viewpoints (sec)	
3 room	111.3	55323	3.4	53.8	28712
4 room	148.3	72353	4.6	61.9	42103

The above results indicate that the overhead of the preferred embodiment is low. It also indicates that substantial computation savings can be achieved when the depth complexity of the models is high. There are a number of advantages to be gained by applying the strategies in the disclosed embodiments. Firstly, they do not have the restrictions encountered by the earlier hidden surface algorithms. For example, they do not preclude the presence of intersecting surfaces or patches in the environment. In addition, surface patches can be curved and the scene need not be static. The method also has very little computational overhead. Because it operates on a z-buffer, it can be easily implemented in hardware.

The preferred embodiment can be further enhanced using heuristics. In applications such as the interactive walkthrough of the interior of a building, the surfaces most likely to form large crisp (non-fuzzy, umbra) regions are those representing walls, floors and ceilings. Therefore, only the crisp regions of these surfaces need to be scan-converted to the fuzzy buffer.

In addition to reducing hidden surface removal computations, the preferred method can also be used in the radiosity method. Currently the radiosity method is very slow, mainly because it needs to compute the visibility of surfaces for a large number of surface points to determine the form-factors. However, by grouping these points and treating them as groups of viewpoints, the preferred embodiment can be used to determine surfaces totally visible to a group of these points. The fuzzy computations for each viewpoint bounding box is apportioned among all viewpoints in the box. Because there would be viewpoint for every surface point, the density of viewpoints in each viewpoint bounding box could be very high. Therefore, the advantage of using the method in the radiosity method can be more than in the normal hidden surface removal. The methods disclosed can also be used in virtual reality applications and other applications where abstract data is utilized. In the future such an application would need to display complex and dynamic environments in real time using the radiosity method. With the addition of a time dimension, the density of viewpoints in each viewpoint bounding box could be an order of magnitude higher than that of the radiosity method. Consequently, the methods disclosed could substantially speed up these applications. Other applications include the manipulation of physical data sue as the generation, processing and display of scientific data such as energy spectra data.

The foregoing describes only a number of embodiments of the present invention and modifications, obvious to those skilled in the art, can be made thereto without departing from the scope of the present invention.

APPENDIX 1

The Detection of Patches Front-Facing to a Viewpoint Bounding Box Referring to FIG. 2, whether a patch PT is facing a viewpoint VP can be determined by the angle A

US 6,618,047 B1

19

between the normal N of the patch PT and the view vector VV . The latter is the vector from the viewpoint VP to the centre of the patch PT . The cosine of the angle is:

$$\begin{aligned}\cos A &= N \cdot (PO - VP) * k \\ &= k(Nx(x' - x) + Ny(y' - y) + Nz(z' - z)) \\ &= k(Nxx' + Nyy' + Nzz' - Nxx - Nyy - Nzz)\end{aligned}$$

where $VP=(x, y, z)$ is the position of the viewpoint, $N=(Nx, Ny, Nz)$ is the unit normal of the patch, and $PO=(x', y', z')$ is the position of the patch centre. k is a positive coefficient which turns the view vector ($VV=PO-VP$) into a unit vector.

If the patch PT is facing the viewpoint bounding box BB , the angle A must be obtuse. Hence cosine A must be negative for every possible viewpoint VP in the box. To check this, the maximum value of the cosine of A is found. If it is positive, then the patch would be back-facing to certain viewpoints in the bounding box.

Since PO and N are constant vectors, the maximization of cosine A requires each of Nx x , Nyy and Nz z to be minimized. Depending on the signs of the components of N , either the maxima or the minima of x , y and z are used. These coordinates correspond to the locations of the corners of the current viewpoint bounding box.

If a patch moves or rotates over time and the viewpoint bounding box contains viewpoints at different times, then the smallest cosine A at each instance of time may be computed and the smallest of them chosen. Patches with bounding volumes inside or intersecting the viewpoint bounding box are always treated as not front-facing. These patches can be detected by an initial bounding volume check.

FIG. 11A shows a cut-plane view of a surface region SU_R front-facing a set of viewpoints VP_7, VP_8, VP_9 and FIG. 11B shows the same surface region front-facing to a viewpoint bounding box BB_R .

Since the front-facing-patches PT_P are facing all possible viewpoints in the box BB_R , the region SU_R never curves back and is always facing these viewpoints. The view projections of the border of the region SU_R therefore always enclose its projection. This property can be used in subsequent fuzzy projection computations.

FIGS. 12A–12C show cut-plane views of projections of surface regions front facing to a group of viewpoints and their respective bounding boxes.

In FIG. 12A if all patches PT_A in a surface region are front facing, the projections SP_A of the boundary edges of the region always surround the projection PE_A of that region.

In FIG. 12B if some surface patches PT_A in a surface region are not front facing, the region may curve back. The projections BP_B of the boundary edges of the region may be inside the projection PE_B of that region.

In FIG. 12C the projection BP_C of the boundary of each front-facing region of a convoluting surface still encloses the projection PE_C of the region.

APPENDIX 2

The Computation of the Fuzzy Extent of Boundary Edges and Patches

As shown in FIG. 2, the projection coordinates $PR(X, Y)$ of a point $PO(x', y', z')$ to a viewpoint $VP(x, y, z)$ on the projection plane PP facing the positive z direction is

$$X = (x' - x) * d / (z' - z) \quad (1)$$

$$Y = (y' - y) * d / (z' - z) \quad (2)$$

20

where d is the shortest distance from the viewpoint to the projection plane facing the z direction.

To maximize X and Y , the minimum values of x and y are used. If X and Y are to be minimize, the maxima of x and y are used. Since z is part of the divisors, whether to use its maximum or minimum value depends on the signs of $(x'-x)$ and $(y'-y)$ after choosing the extreme values of x and y . Since the edges of the viewpoint bounding box are parallel to the axes, viewpoints having these extreme x , y and z values are at the corners of the box.

According to equations (1) and (2), the points which have the extreme values in x' , y' and z' would yield the maximum and minimum projections. However, these points might not be on the current projecting entity, which can be a boundary edge or a patch.

To overcome the problem, the bounding volume of the entity is used to approximate that entity. Its projections can be used as an estimation as they enclose the projections of that entity.

Sometimes an entity may be subdivided into sub-entities. Its position and shape may also be non-static over time or uncertain. Hence the bounding volume of an entity should be a volume that encloses all the space it is likely to occupied, and all the bounding volumes of its sub-entities that are likely to be considered.

Hence the bounding volume of an entity should enclose all its sub-entities that are likely to be considered in the computations. Following the derivations above, the maxima of x' and y' and the minima of x and y are used to maximize the projections. Whether to maximize or minimize $(z'-z)$ depends on the signs of $(x'-x)$ and $(y'-y)$. If these terms are positive, the term should be minimized. Hence the smallest z' and the largest z are used. Otherwise, the largest z' and the smallest z are instead chosen.

The finding of the minimum projections is the opposite of maximization. Hence the decisions to minimize or to maximize the individual terms are accordingly changed.

Having obtained the projections which give the maximum and minimum X and Y values, these projections are mapped to the fuzzy projection plane FP whose normal is toward the z direction.

A rectangle on the plane with these projections as corners and with edges parallel to the edges of the corresponding fuzzy projection face FF can be defined. They are either rounded up or down to create another region which contains the original rectangle and all the pixels under it. This is the fuzzy extent EF of the patch on the fuzzy projection plane FP .

By rotating the axes, the same equations (1) and (2) can be used for projection planes facing other directions if the terms in equations (1) and (2) are accordingly permuted. The fuzzy extents corresponding to these planes can then be similarly obtained.

If a patch is within the viewpoint bounding box, then for some viewpoints z' is smaller than or equal to z in equations (1) and (2). The fuzzy extents would then extend to infinity in the directions which can be determine by the signs of the denominators and numerators in these equations.

APPENDIX 3

The Computation of the Non-Fuzzy Regions

To compute the non-fuzzy region, it is first necessary to obtain its relationship with the fuzzy boundary region and one of the projection regions.

Consider two viewpoints in the current viewpoint bounding box and a point on a fuzzy projection plane. Assume that

US 6,618,047 B1

21

the projection of the front-facing sub-surface at one viewpoint maps on that point and at the other viewpoint does not map on that point.

Because the change of projection status can only occur at the projections of boundary edges, on a curve joining the two points there must exist another viewpoint such that the projection of a boundary edge from it would map on the same point on the fuzzy projection plane. Since this curve can be totally inside the current viewpoint bounding box, that point must be within the fuzzy region of the boundary edges for that bounding box.

If a point on a fuzzy projection plane is within the mapping of a projection region of the front-facing sub-surface, and if it is outside the fuzzy region of the boundary edge, then the point is always within the projection regions of the sub-surface for all the viewpoints in the current viewpoint bounding box. The point is by definition inside the non-fuzzy region. Since the fuzzy boundary region is always within the area containing the fuzzy extents of boundary edges, the point is still inside the non-fuzzy region if the former is approximated by the latter.

Based on the above, the approximated non-fuzzy region can be obtained by finding the projection region of the front-facing sub-surface from a viewpoint in the current viewpoint bounding box, projecting it to the fuzzy region, and subtracting all its area intersected with the fuzzy extents of the boundary edges. A modified scan line algorithm may be used for this task.

First, an arbitrary viewpoint in the current viewpoint bounding box is selected. Usually the centre of the bounding box is chosen. This viewpoint is called the chosen viewpoint. The projection coordinates of the edges in the front-facing sub-surface from this viewpoint are found.

Since the projection coordinate system and the fuzzy projection coordinate system have one-to-one mapping, the mapping of the former to the latter is trivial.

This is followed by the computations of the fuzzy extents of the patches described in Appendix 2.

An illustration of such an arrangement is shown in FIG. 8A where two patches A and B intersect a scan line SL and have respective fuzzy extents EF_A and EF_B of their boundary edges.

Two active edge lists are maintained during the processing of scan lines. The first list contains the active edges of the patches. The second list contains the pairs of active vertical edges of the fuzzy extents of boundary edges.

In FIG. 8A, the first active edge list is:

A1 A2 B1 B2

the second active edge list is:

E1 E2 E3 E4

During the processing of successive scan lines, entries in the list are added when they become active in the current scan line. They are deleted from the list when the last scan lines they are active have been processed.

A scan line array with as many elements as the number of horizontal elements in the current fuzzy array is maintained. Each element of the array contains a depth field and a boolean field called the overlap indicator. The depth field is used to store the depth of the patch whose fuzzy extent surrounds the fuzzy element. The overlap indicator indicates whether the element is under the non-fuzzy region. The depth fields are initialized to negative infinity. The overlap indicators are set to zero.

The pairs of edges in the first active edge list are then processed. The span of each pair of these edges represents

22

the segment of a patch in the current scan line. The overlap indicators of the elements within the span are set to one. The value stored in the depth field of each of these elements is compared with the depth of the patch with respect to the chosen viewpoint. It may be approximated by the maximum depth from the viewpoint to the patch or the patch bounding box. If the former is smaller, it is replaced by the latter.

The computation of the segments of patches requires the finding of the intersections between the current scan line and the active edges. This computation can be avoided by using the leftmost or rightmost position of each edge depending on whether it is at the left or right side of the span. This inaccuracy is allowed because it is always offset by the fuzzy extents of the boundary edges.

The pairs of active edges in the second active edge list are then accessed. The span of each pair represents the segment of the fuzzy extent of a boundary edge in the current scan line. All the overlap indicators of the elements within the span are reset to zero.

After all the records in the active edge lists have been processed, the elements in the scan line array where the overlap indicators are one are within the segment of the non-fuzzy region in the current scan line. The depth fields contain the maximum possible depth of the region in the viewing directions represented by these elements.

The depth stored in each element of the array is compared with the depth field of the corresponding element in the fuzzy buffer. If the latter is larger, it is replaced by the former.

Returning to the FIGS., FIG. 8B shows the scan line array after scan-conversion of the active edges of patch A. FIG. 8C shows the scan-conversion of the active edges of patch B. FIG. 8D shows the scan line array after the elements within the fuzzy extents EF_A and EF_B of the boundary edge have been reset.

APPENDIX 4

The Scan-conversion of the Fuzzy Regions

To obtain the fuzzy regions, a modified scan line algorithm to that of Appendix 3 is used.

An active edge list is maintained during the processing of scan lines. The list contains pairs of vertical edges of the fuzzy extents of patches active in the current scan line. During the processing of successive scan lines, entries in the list are added when they become active in the current scan line. An entry is deleted from the list when the last scan line it is active has been processed.

For the arrangement of FIG. 9A (similar to FIG. 8A, but for fuzzy regions), the active edge list is:

E1 E2 E3 E4.

A scan line array with as many elements as the number of horizontal elements in the current fuzzy array is maintained. Each element of the array is a boolean field called the overlap indicator which indicates whether the element is under the fuzzy region.

For each scan line, the entries in the active edge list are accessed. The span between each pair represents the segment of the fuzzy extent of a patch in the current scan line. All the overlap indicators of the elements within the span are set to one.

After the processing of the entries in the active edge list. The array elements whose overlap indicators are one now contain the segment of the fuzzy region in the current scan line.

FIG. 9B shows the state of the scan line array after the elements within the fuzzy extent EF_A of patch A have been updated.

FIG. 9C shows the state of the scan line array after the elements within the fuzzy extent EF_B of patch B have been updated.

APPENDIX 5

Using a Cache to Speed UD the Access and Writing of Fuzzy Elements Within the Fuzzy Extent of a Patch

The fuzzy buffer methods described herein require the reading and writing of the fuzzy elements within the fuzzy extent of each patch. Since there are usually substantial overlaps between the fuzzy extents of adjacent patches, the access and update of these elements can be reduced by storing their information in high speed cache.

The cache may be in many forms. One example, shown in FIG. 10, is to use a cache buffer CB and two cross-reference tables CR_1 and CR_2 . The cache buffer CB is a 2D array, in which each element contains fields which are the mirror of the fields on a fuzzy buffer element that need to be read or written. The cross-reference tables CR_1 and CR_2

respectively contain as many elements as the number of rows and columns in the 2D array. Depending on the array it is in, each element in the table contains either a row (CR_1) or a column (CR_2) field which is used to cross-reference the rows and columns of the 2D array with the rows and columns of the current fuzzy array FA. For each element in the cache, the cross-reference tables CR_1 and CR_2 contain a reset switch to indicate whether it has been reset.

Initially, all the reset switches are initialized to one. The centre of the cache buffer CB is mapped to the centre of the fuzzy extent of the first patch to be read. After the fuzzy elements within the fuzzy extent of the first patch are read, information in these elements are stored in the cache buffer CB elements according to the mapping. The mappings of rows and columns in the cache are updated into the cross reference table CR_1 and CR_2 , respectively. The reset switch in the updated elements in these table and the cache buffer CB are set to zero. Before reading the fuzzy elements within the fuzzy extent of the second patch, data in the cache buffer CB is first read. If an element has already been stored in the cache buffer CB, no access to the cache buffer CB to obtain its information is necessary.

To maximize the efficiency of the cache, any adjacent patch is accessed first. After the processing of a series of patches, there may be a patch whose mapping covers the boundary of the cache. The affected rows or columns are wrapped around. The rows at the opposite edges become the next rows or columns for the patch.

Before the writing of the patch data into the cache buffer CB each element in the cross-reference table CR_1 and CR_2 within the rows of the columns of the current patch mapping are checked. If the reset switch is on or if the mapping value conforms to the current mapping the whole row or column of cache elements have not been used and there is no problem in using them. Otherwise the row or the column needs to be reclaimed. The original data in the whole row or column of cache elements are written back to the fuzzy buffer and then initialized. The affected element in the cross-reference table is updated with the new mapping. After that the patch data can then be written into these newly-reclaimed cache elements.

The writing from the cache buffer CB to the fuzzy buffer can be performed in burst mode. Hence when such task occurs several adjacent rows or columns of the cache elements may be written to the fuzzy buffer and re-initialized even though some of the elements may not need to be re-claimed.

APPENDIX 6

Direct Computation of the Fuzzy Region of an Edge

Assume that the equation of the edge is expressed as:

$$x=az+b$$

$$y=cz+d$$

2. Assume that all the coordinates of a viewpoint P in the current viewpoint bounding box is $(x_0 \ y_0 \ z_0)$.

3. The image projection of a point T $(x \ y \ z)$ on the edge with respect to P is

$$X = D \frac{x - x_0}{z - z_0}$$

$$Y = D \frac{y - y_0}{z - z_0}$$

4. Since $(x \ y \ z)$ and $(x_0 \ y_0 \ z_0)$ are uncertain the region comprising all possible values of $(X \ Y)$ forms an area which is the fuzzy region of the edge.

5. By substituting (2) into (4) and rearranging

$$z = \frac{Dd - Dy_0 + Yz_0}{Y - Dc}$$

If Y is known the extreme of z can be found by substituting the appropriate extreme of y_0 and z_0 into the above equation. All points on the edge having z within these extreme could project at Y from viewpoints in the current viewpoint bounding box.

6. Substituting (1) into (3)

$$X = D \frac{az + b - x_0}{z - z_0}$$

Notice that in (6) there is no local extremum of X for changing z. Hence by assigning the extreme of z found from section 5 and the appropriate extreme values of x_0 and z_0 into (6) the two extreme of X can be found. These values correspond to the maximum and minimum possible X for a particular value of Y.

7. A scan line approach can be used to approximate the fuzzy region of the edge. Substituting (2) into (4) and rearranging

$$Y = D \frac{cz + d - y_0}{z - z_0}$$

By assigning the appropriate extreme values of $z \ y_0$ and z_0 into (7) the maximum and minimum values of Y are obtained. These values are rounded up and down respectively to cover all scan lines the edge may project onto. Each of these scan lines corresponds to a value of Y. From section 5 and 6 the maximum and minimum of X can be found. The area within these values is the scan line segment of the fuzzy region of the edge. The corresponding segment of the expanded fuzzy region of that edge is the further rounding of the segment to fully cover all the grid cells it projects cover.

The above technique computes the expanded fuzzy regions by scan-converting the segments of the regions on the X scan lines. For edges more horizontal than vertical the

US 6,618,047 B1

25

expanded fuzzy regions may be more accurately computed by scan-converting the segments of the regions on the Y scan lines.

This is shown in FIG. 14 where the fuzzy region takes on a scan-line shape about the fuzzy region of the edge. This is substantially smaller than the fuzzy extent of the edge.

APPENDIX 7

Scan-conversion of Patches on the Hemicube Pyramid

This appendix describes an efficient technique for scan-converting a patch on a plane which contains cells with different resolutions. It can be used to compute the form-factor of patches in the radiosity method with or without the use of the fuzzy projection techniques.

The technique uses a data structure called the hemicube pyramid. Each level of the pyramid contains a set of 2D arrays. Each array corresponds to a level and a projection face. Each element of the array corresponds to a pixel on the plane. Note that the hemicube may be a prism rather than a cube and not all its planes need to be active.

An element in the array is called a pyramid element. In addition to the information needed in an original hemicube element it contains a sub-division indicator and a pointer. The sub-division indicator is initially set to OFF. The indicator becomes active if its level is less than the required accuracy level of patches projected onto it.

The pointer is used to link a pyramid element with one of the pyramid elements at the higher accuracy level and within its region. To enable more accurate sampling the elements to be linked are usually chosen such that an element accessed by the pointers from an element at the lower accuracy level is as close to the centre of the latter as possible.

The organization of the hemicube pyramid corresponding to a face of the hemicube is shown in FIG. 15.

After the initialization of the hemicube pyramid patches are accessed in decreasing order of accuracy and scan-converted to the hemicube pyramid. This order of access ensures that patches are scan-converted in resolutions equal to or finer than their own optimal projection resolutions and the optimal projection resolutions of patches they might occlude.

For each projecting patch the hemi-cube projection resolution corresponding to its accuracy level is found. Each hemicube pyramid element belonging to that resolution and under The projection of that patch is accessed.

If the accessed element is already at the most accurate level, or if its sub-division indicator is not on, the ID of the current patch is written into the current element if that patch is closer than the so-far closest patch stored in the element, as in the normal hemicube scan-conversion. If this update occurs, all the ancestors of the current element until the element whose sub-division indicator has already been set to on are accessed. The sub-division indicator of each accessed element is set to on.

Note that the recursive access of ancestor elements can instead be carried out for all elements under the patch projection immediately after they have been found. However, if the possibility that patches are invisible is high, carrying out the recursion after determining that the patch is closer is more efficient as it does not need to be applied on elements if that patch is hidden.

If the sub-division indicator of the current element is ON, patches requiring more accurate form-factor evaluation have

26

been projected onto the element. Therefore, each of its daughter elements under the projection of the current patch is accessed and the above computations are repeated.

After the scan-conversion of all projecting patches, the elements in all the levels are scanned. From the projection information stored in these elements, the form factors of the patches are computed.

If highly accurate form-factor computations and hence very high hemi-cube resolutions are required, relatively large amounts of memory are needed to store the hemi-cube arrays. The initialization and scanning of these arrays for form-factor accumulation is also costly.

Since patches requiring very fine projections are relatively few, the above problem can be overcome by using sparse matrices to store active (sub-divided) elements corresponding to high resolution levels. Each matrix is a one-dimensional array. Active elements are stored in it at positions determined by a hashing function parameterized by pixel positions.

High accuracy form-factor computations can also be achieved using ray-tracing in conjunction with hemicube computations. All the patches are still scan-converted to the hemicube pyramid or the normal hemicube. However, the scan-conversion of patches requiring high accuracy in form-factor computations is only used to determine how other patches are blocked by them. Instead, their form-factors are determined by tracing rays from them.

The content of the hemicube pyramid can be used to speed up the ray-tracing. During the scan-conversion of a patch that is also ray-traced, patches stored in the buffer that are closer to it are determined. When ray-tracing it, these patches may be tested with the rays first as they are more likely to block these rays.

APPENDIX 8

References

1. Aggarwal J. K., "Dynamic Scene Analysis," in "Image Sequence Processing and Dynamic Scene Analysis," Huang, T. S. (Eds.) 1981, Springer-Verlag, pp. 40-73.
2. Appel A. "The Notion of Quantitative Invisibility and the Machine Rendering of Solids", Proc. of the ACM National Conference, 1967, Thompson Books, Washington D.C., pp.387-393.
3. Baum, D. R., Rushmeier H. E., Winget J. M., "Improving Radiosity Solutions through the use of Analytically Determined Form-factors", Computer Graphics, Vol 23 No. 3, 1989, pp. 325-334.
4. Bradler, N., Tsotsos, J. K., (Eds) "Motion: Representation and Perception," North-Holland, 1986.
5. Catmull, E. "A Subdivision Algorithm for Computer Display of Curved Surfaces", PhD Thesis, Report UTEC-CSc-74, University of Utah, 1975.
6. Chang, S. S. L., Zadeh, L. A., "On Fuzzy Mapping and Control," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-2, No. 1, January 1972, pp. 30-34.
7. Cohen, M. F., D. P. Greenberg, "The Hemi-Cube: A Radiosity Solution for Complex Environment," SIGGRAPH 85, pg. 31-40.
8. Cohen, M. F., Chen, S. E., Wallace, J. R., Greenberg, D. P., "A Progressive Refinement Approach to Fast Radiosity Image Generation," SIGGRAPH 88, pg. 75-84.
9. Crocker, G. A., "Invisibility Coherence for Faster Scan-Line Hidden Surface Algorithms," SIGGRAPH '84, pp. 315-321.
10. Fournier, A., Fussell, D., "On the power of the Frame Buffer," ACM Transactions of Graphics, April 1988, pp. 103-128.

US 6,618,047 B1

27

11. Fuch, H., Kedem, Z., Naylor, B. F., "On Visible Surface Generation by A Priority Tree Structures," SIGGRAPH'80, pp. 124-133.
12. Gordon, D., "Front-to-Back Display of BSP Trees," IEEE CODA, September 1991, pp. 79-85.
13. Haines, E. A., Wallace, J. R., "Shaft Culling for Efficient Ray-Traced Radiosity," unpublished paper, July 1991.
14. Hornung, C. "A Method for Solving the Visibility Problem", IEEE Computer Graphics and Applications, July 1984, pp. 26-33
15. Hubschman, H., Tucker, S. W., "Frame-to-Frame Coherence and the Hidden Surface Computation: Constraints for a Convex World," SIGGRAPH 81, 45-54.
16. Impel, D. S., Cohen, M. F., "A Radiosity Method for Non-Diffuse Environments," Computer Graphics, Vol. 4, 1986, pp. 133-142.
17. Jain, R., Haynes S., "Imprecision in Computer Vision," in "Advances in Fuzzy Sets, Possibility and Applications," Wang, P. (Ed), Plenum Press, 1983, pp. 217-236.
18. Kanatani, K., "Group-Theoretical Methods in Image Understanding," Springer Verlag, 1990.
19. Kaufmann, A., "Theory of Fuzzy Subsets, Vol. 1, Fundamental Theoretical Elements", Academic Press, 1975.
20. Ligomenides, P. A., "Modeling Uncertainty in Human Perception," in "Uncertainty in Knowledge-Based Systems," Bouchon, B., Yager, R. (Eds), Springer Verlag, 1986, pp. 337-346.
21. Lim, H. L., "Fast Hidden Surface Removal Through Structural Analysis and Representation of Objects and Their Contours," CGI'87, Japan, 1987, pp. 75-88.
22. Marks, J., Walsh, R., Christensen, J., Friedell, M., "Image and Intervisibility Coherence in Rendering", Proceedings of Graphics Interface '90, Toronto, Ontario, May 1990, pp. 17-30.
23. Recker, R. J., George D. W., Greenberg D. P., "Acceleration Techniques for Progressive Refinement Radiosity," Proceedings, 1990 Symposium on Interactive 3D Graphics, Snowbird, Utah, Computer Graphics, pp. 59-66.
24. Ruspini, E. H., "A New Approach to Clustering," Information Control, Vol. 15, pp. 22-32.
25. Siedlecki, W., Siedlecka, K., Sklansky, J., "Mapping Techniques for Exploratory Pattern Analysis," in "Pattern Recognition and Artificial Intelligence," Gelsema, E. S., Kanal L. N., (Eds), Elsevier Science, 1988, pp. 277-299.
26. Sillion, F., Puech, C., "A General Two-Pass Method Integrating Specular and Diffuse Reflection," SIGGRAPH 1989, pg. 335-344.
27. Subbarao, M., "Interpretation of Visual Motion: A Computational Study," Pitman, London, 1988.
28. Sutherland, I. E., Sproull, R. F., Schumacker, R. A., "A Characterization of Ten Hidden Surface Algorithms", Computing Surveys, Vol. 6, No. 1, 1974, pp. 1-55
29. Teller, S. J., "Visibility Preprocessing for Interactive Walkthrough," Computer Graphics, Vol. 25, No. 4, July 1991.
30. Wallace, J. R., Elmquist, K. A., Haines, E. A., "A Ray Tracing Algorithm for Progressive Radiosity", Computer Graphics, Volume 23, No. 3, July 1989, pp. 315-324.
31. Yee, L., "Spatial Analysis and Planning under Imprecision", North-Holland, 1988.
32. Zadeh, L. A., "Fuzzy Sets", Information and Control, Vol. 8, 1965, pp. 338-353.

What is claimed is:

1. A method of reducing the visibility related computations in 3-D computer graphics, the visibility related com-

28

putations being performed on 3-D surfaces or their sub-elements, or a selected set of both, the method comprising:

- identifying grid cells which are under or related to the projections or extents of projections associated with at least one of said 3-D surfaces or their sub-elements;
- comparing data associated with said at least one of 3-D surfaces or their sub-elements with stored data associated with the grid cells;
- determining which of said at least one of 3-D surfaces or their sub-elements is always invisible or always visible to a viewpoint or a group of viewpoints by projection based computations prior to a visibility computation; and
- ignoring said determined at least one of the 3-D surfaces or their sub-elements during said visibility computation.

2. The method of claim 1 wherein the projection based computations comprise:

- defining one or more projection planes for generating projections with respect to a viewpoint or a group of viewpoints;

- identifying regions on the projection planes that are or are related to the projections or the extents of projections associated with selected 3-D surfaces.

3. The method of claim 2 wherein said identifying step comprises dividing each projection plane into one or more grids; and storing the data of said projections on the grids in computer storage.

4. The method of claim 3 wherein said grid or grids are either regular or irregular.

5. The method of claim 3 wherein the structure of said computer memory is based on a z-buffer or a quadtree.

6. The method of claim 3 wherein some or all of said data are or are related to the depths of said 3-D surfaces or their sub-elements.

7. The method of claim 3 wherein some or all of said data are or are related to the shortest depths of said 3-D surfaces or their sub-elements.

8. The method of claim 3 wherein some or all of said data are or are related to the largest depths of said 3-D surfaces or their sub-elements.

9. The method of claim 1 wherein at least one of the data to be compared with said stored data is a value related to the depth value, the largest depth value, or the smallest depth value related to one of said 3-D surfaces or said sub-elements.

10. The method of claim 1 wherein said visibility related computations are speeded up by using cache memory.

11. A method of reducing a step of visibility computations in 3-D computer graphics from a perspective of a viewpoint, the method comprising:

- computing, before said step and from said perspective, the visibility of at least one entity selected from 3-D surfaces and sub-elements of said 3-D surfaces, wherein said computing step comprises at least a comparison between a pair of depth-related numbers to determine which of the surfaces or the sub-elements of the surfaces associated with said numbers is closer to said viewpoint and said 3-D surfaces are arranged in a hierarchy represented by patches in varying levels of details; and

- skipping, at said step, at least an occlusion relationship calculation for at least one entity that has been determined to be invisible in said computing step.

12. A method of reducing a step of visibility computations in 3-D computer graphics from a perspective of a viewpoint, the method comprising:

US 6,618,047 B1

29

computing, before said step and from said perspective, the visibility of at least one entity selected from 3-D surfaces and sub-elements of said 3-D surfaces, wherein said computing step comprises:

employing at least one projection plane for generating 5
projections with said selected set of 3-D surfaces and said sub-elements with respect to said perspective;
identifying regions on said at least one projection plane, wherein said regions are related to the pro- 10
jections associated with said selected 3-D surfaces, said sub-elements, or bounding volumes of said 3-D surfaces or said sub-elements;
updating data related to said regions in computer stor-
age; and

deriving the visibility of at least one of said 3-D surfaces 15
or said sub-elements from the stored data in said computer storage; and
skipping, at said step of visibility computations, at least an occlusion relationship calculation for at least one entity that has been determined to be invisible in said 20
computing step.

13. The method of claim 12, wherein at least one of said bounding volumes is a volume containing a 3-D surface, or a volume containing a 3-D surface and at least one of the sub-elements of said 3-D surface. 25

14. The method of claim 12, wherein at least one of said bounding volumes defines a space containing all likely occurrence of said 3-D surface, or said 3-D surface and at least one of said sub-elements. 30

15. The method of claim 12, wherein said identifying step comprises: 35

dividing each projection plane into at least one grid; and
identifying the cell or cells of said at least one grid which are related to each of said regions or extents of said regions. 40

16. The method of claim 15, wherein said at least one grid is regular. 45

17. The method of claim 16, wherein the structure of said computer storage is based on a z-buffer. 50

18. The method of claim 15, wherein said at least one grid is irregular. 55

19. The method of claim 18, wherein the structure of said computer storage is based on a quadtree. 60

20. The method of claim 12, wherein said updating step comprises: 65

performing a depth comparison test between data associ-
ated with each of said identified cell or cells with said stored data in an element of said computer storage, wherein said element is the one associated with said cell; and 50

if said test indicates that update is necessary for said cell, writing said data associated with said cell into said associated element.

21. The method of claim 20, wherein the data to be 55
compared with said stored data is related to the depth of one of said 3-D surfaces or one of said sub-elements of said 3-D surfaces, wherein the projection region or the extent of projection region of said one of said 3-D surfaces or said one of said sub-elements of said 3-D surfaces is related to said cell. 60

22. The method of claim 20, wherein the data to be compared with said stored data is related to the shortest depth of said 3-D surface or said sub-element.

23. The method of claim 20, wherein the data to be 65
compared with said stored data is related to the largest depth of said 3-D surface or said sub-element.

30

24. The method of claim 20, wherein the data to be compared with said stored data is related to the bounding volume associated with said 3-D surface or said sub-element.

25. The method of claim 24, wherein at least one of said bounding volume is a volume containing a 3-D surface, or a volume containing a 3-D surface and at least one of the sub-elements of said 3-D surface.

26. The method of claim 24, wherein at least one of said bounding volumes defines a space containing all likely occurrence of said 3-D surface, or said 3-D surface and at least one of said sub-elements.

27. The method of claim 12, wherein said computer storage is accelerated by cache memory.

28. A method of reducing visibility computations in 3-D computer graphics, the method comprising:

computing the visibility of a set of entities selected from 3-D surfaces and sub-elements of said 3-D surfaces from the perspective of a group of a plurality of viewpoints, wherein said computing step comprises at least a comparison between a pair of depth-related numbers to determine which of the surfaces or the sub-elements of the surfaces associated with said num-
bers is closer to said group of a plurality of viewpoints and said 3-D surfaces are arranged in a hierarchy represented by patches in varying levels of details; and
skipping at least an occlusion relationship calculation at subsequent steps of visibility computations for at least one entity that has been determined to be invisible in said computing step, wherein said visibility computa-
tions in each step is from the perspective of each viewpoint from said group, or from the perspective of each subset of viewpoints from said group. 30

29. A method of reducing visibility computations in 3-D computer graphics, the method comprising: 35

computing the visibility of a set of entities selected from 3-D surfaces and sub elements of said 3-D surfaces from the perspective of a group of a plurality of viewpoints, wherein said computing step comprises:
employing at least one projection plane for generating projections with said selected set of 3-D surfaces and said sub-elements with respect to the perspective of a group of a plurality of viewpoints; 40

identifying regions on said at least one projection plane, wherein said regions are related to the pro-
jections associated with said selected 3-D surfaces, said sub-elements, or bounding volumes of said selected 3-D surfaces or said sub-elements from said perspective; 45

updating data related to said regions in computer stor-
age; and

deriving the visibility of at least one of said 3-D surfaces or said sub-elements from the stored data in said computer storage; and

skipping at least an occlusion relationship calculation at subsequent steps of visibility computations for at least one entity that has been determined to be invisible in said computing step, wherein said vis-
ibility computations in each step is from the perspec-
tive of each viewpoint from said group, or from the perspective of each subset of viewpoints from said group. 50

30. The method of claim 29, wherein at least one of said bounding volumes is a volume containing a 3-D surface, or a volume containing a 3-D surface and at least one of the sub-elements of said 3-D surface.

31. The method of claim 29, wherein at least one of said bounding volumes defines a space containing all likely

US 6,618,047 B1

31

occurrence of said 3-D surface, or said 3-D surface and at least one of said sub-elements.

32. The method of claim 29, wherein said identifying step comprises:

dividing each projection plane into at least one grid; and
identifying the cell or cells of said at least one grid which
are related to each of said regions or extents of said
regions.

33. The method of claim 32, wherein said at least one grid is regular.

34. The method of claim 33, wherein the structure of said computer storage is based on a z-buffer.

35. The method of claim 32, wherein said at least one grid is irregular.

36. The method of claim 35, wherein the structure of said computer storage is based on a quadtree.

37. The method of claim 29, wherein said updating step comprises:

performing a depth comparison test between data associated with each of said identified cell or cells with said stored data in an element of said computer storage, wherein said element is the one associated with said cell; and

if said test indicates that update is necessary for said cell, writing said data associated with said cell into said associated element.

38. The method of claim 37, wherein the data to be compared with said stored data is related to the depth of one of said 3-D surfaces or one of said sub-elements of said 3-D surfaces, wherein the projection region or the extent of projection region of said one of said 3-D surfaces or said one of said sub-elements of said 3-D surfaces is related to said cell.

39. The method of claim 37, wherein the data to be compared with said stored data is related to the shortest depth of said 3-D surface or said sub-element.

40. The method of claim 37, wherein the data to be compared with said stored data is related to the largest depth of said 3-D surface or said sub-element.

41. The method of claim 37, wherein the data to be compared with said stored data is related to the bounding volume associated with said 3-D surface or said sub-element.

42. The method of claim 41, where at least one of said bounding volume is a volume containing a 3-D surface, or a volume containing a 3-D surface and at least one of the sub-elements of said 3-D surface.

43. The method of claim 41, wherein at least one of said bounding volumes defines a space containing all likely occurrence of said 3-D surface, or said 3-D surface and at least one of said sub-elements.

44. The method of claim 29, wherein said computer storage is accelerated by cache memory.

45. A method of reducing visibility computations in 3-D computer graphics, the method comprising:

computing the visibility of a set of entities selected from 3-D surfaces and sub-elements of said 3-D surfaces from the perspective of a group of at least one viewpoint, wherein said computing step comprises at least a comparison between a pair of depth-related numbers to determine which of the surfaces or the sub-elements of the surfaces associated with said numbers is closer to said group of at least one viewpoint and said 3-D surfaces are arranged in a hierarchy represented by patches in varying levels of details; and
skipping at least an occlusion relationship calculation at each of subsequent step or steps of visibility

32

computations, wherein said visibility computations in each of said step or steps is from the perspective of each viewpoint from said group, or from the perspective of a subset of viewpoints from said group, for at least one entity that has been determined to be visible in said computing step.

46. A method of reducing visibility computations in 3-D computer graphics, the method comprising:

computing the visibility of a set of entities selected from 3-D surfaces and sub-elements of said 3-D surfaces from the perspective of a group of at least one viewpoint, wherein said computing step comprises:

employing at least one projection plane for generating projections with said selected set of 3-D surfaces and said sub-elements with respect to said perspective of a group of at least one viewpoint;

identifying regions on said at least one projection plane; wherein said regions are related to the projections associated with said 3-D surfaces, said sub-elements, or bounding volumes of said 3-D surfaces or said sub-elements from said perspective;

updating the data related to said regions in computer storage; and

deriving the visibility of at least one of said 3-D surfaces or said sub-elements from the data stored in said computer storage, and

skipping at least an occlusion relationship calculation at each of subsequent step or steps of visibility computations, wherein said visibility computations in each of said step or steps is from the perspective of each viewpoint from said group, or from the perspective of a subset of viewpoints from said group, for at least one entity that has been determined to be visible in said computing step.

47. The method of claim 46, wherein at least one of said bounding volumes is a volume containing a 3-D surface, or a volume containing a 3-D surface and at least one of the sub-elements of said 3-D surface.

48. The method of claim 46, wherein at least one of said bounding volumes defines a space containing all likely occurrence of said 3-D surface, or said 3-D surface and at least one of said sub-elements.

49. The method of claim 46, wherein said identifying step comprises:

dividing each projection plane into at least one grid; and
identifying the cell or cells of said at least one grid which are related to each of said regions or extents of said regions.

50. The method of claim 49, wherein said at least one grid is regular.

51. The method of claim 50, wherein the structure of said computer storage is based on a z-buffer.

52. The method of claim 49, wherein said at least one grid is irregular.

53. The method of claim 52, wherein the structure of said computer storage is based on a quadtree.

54. The method of claim 46, wherein said updating step comprises:

performing a depth comparison test between data associated with each of said identified cell or cells with said stored data in an element of said computer storage, wherein said element is the one associated with said cell; and

if said test indicates that update is necessary for said cell, writing said data associated with said cell into said associated element.

US 6,618,047 B1

33

55. The method of claim 46, wherein said computer storage is accelerated by cache memory.

56. A method of reducing visibility computations in 3-D computer graphics with respect to a viewpoint, the method comprising:

employing at least one projection plane for generating projections with respect to a perspective of said viewpoint;

dividing said at least one projection plane into at least one grid, wherein said at least one grid is irregular;

identifying, from said perspective, grid cells which are related to the projections or extents of the projections, said projections are related to 3-D surfaces, sub-elements of said 3-D surfaces, bounding volumes of said 3-D surfaces, or bounding volumes of said sub-elements, wherein said identified grid cells are represented by a data structure based on a quadtree;

computing visibility of at least one of said surfaces or said sub-elements from said perspective, based on information derived from the identified grid cells; and

ignoring, in a subsequent step of visibility computations, at least one of said 3-D surfaces or said sub-elements that has been determined to be invisible in said computing step, wherein said step is with respect to said perspective.

57. A method of reducing visibility computations in 3-D computer graphics with respect to a viewpoint, the method comprising:

employing at least one projection plane for generating projections with respect to a perspective of said viewpoint;

dividing said at least one projection plane into at least one grid; wherein said at least one grid is regular;

identifying, from said perspective, grid cells which are related to the projections or extents of the projections; said projections are related to 3-D surfaces, sub-elements of said 3-D surfaces, bounding volumes of said 3-D surfaces, or bounding volumes of said sub-elements, wherein said identified grid cells are represented by a data structure based on a z-buffer;

computing visibility of at least one of said surfaces or said sub-elements from said perspective, based on information derived from the identified grid cells; and

ignoring, in a subsequent step of visibility computations, at least one of said 3-D surfaces or said sub-elements that has been determined to be invisible in said computing step; wherein said step is with respect to said perspective.

58. The method of claim 57 further comprising, after the step of identifying grid cells, updating data related to said 3-D surfaces or said sub-elements in a computer storage related to said identified grid cells.

59. The method of claim 58, wherein the data to be compared with the data of said computer storage are related to the depths of said 3-D surfaces or said sub-elements.

60. The method of claim 56, wherein the data to be compared with the data of said computer storage are related to the shortest depths of said 3-D surfaces or said sub-elements.

61. The method of claim 58, wherein the data to be compared with the data of said computer storage are related to the largest depths of said 3-D surfaces or said sub-elements.

62. The method of claim 58, wherein at least some of the data to be compared with the data of said computer storage are related to the bounding volumes of said 3-D surfaces or said sub-elements.

34

63. The method of claim 58, wherein the computer storage is accelerated by cache memory.

64. A method of reducing the visibility related computations in 3-D computer graphics, said visibility related computations being performed on 3-D surfaces or their sub-elements, or a selected set of both, said visibility related computations is from the perspective of a viewpoint, the method comprising:

prior to a visibility computation, identifying grid cells which are under or related to at least one projection or at least one extent of projection that is associated with at least one of said 3-D surfaces or their sub-elements, said at least one projection or at least one extent of projection is from the perspective of said viewpoint;

comparing the data associated with at least one of said 3-D surfaces, their sub-elements, bounding volumes of said 3-D surfaces, or bounding volumes of said sub-elements with the stored data associated with the grid cells;

determining which of said at least one of said 3-D surfaces, or sub-elements is invisible to said viewpoint; and

if said at least one of said 3-D surfaces or sub-elements is determined to be invisible to said viewpoint, ignoring the entity or entities during said visibility computation.

65. A method in 3-D computer graphics for processing the visibility of 3-D surfaces before a subsequent step of visibility computations, said method comprising:

for each selected 3-D surface or sub-element of a 3-D surface, identifying grid cells on a projection plane which are under or related to a projection associated with said selected 3-D surface or sub-element, said projection and said subsequent step of visibility computations are from the perspective of a same viewpoint; for each of said grid cells, accessing the corresponding z-buffer element; and

computing the visibility of the part of said selected 3-D surface or sub-element that projects onto said each of said grid cells by comparing the depth-related data stored in said corresponding z-buffer element with the depth-related value associated with said part.

66. The method of claim 65, further includes the step of using the visibility information computed for said selected 3-D surface or sub-element to reduce the computation for said selected 3-D surface or sub-element in said subsequent step of visibility computations.

67. A method in 3-D computer graphics for processing the visibility of 3-D surfaces before a subsequent step of visibility computations, said method comprising:

for each selected 3-D surface or sub-element of a 3-D surface, identifying grid cells on a projection plane which are under or related to a projection associated with the bounding volume of said selected 3-D surface or sub-element, said projection and said subsequent step of visibility computations are from the perspective of a same viewpoint;

for each of said grid cells, accessing the corresponding z-buffer element; and

computing the visibility of the part of the bounding volume that projects onto said each of said grid cells by comparing the depth-related data stored in said corresponding z-buffer element with the depth-related value associated with said part.

68. The method of claim 67, further includes the step of using the visibility information computed for said selected 3-D surface or sub-element to reduce the computation for said selected 3-D surface or sub-element in said subsequent step of visibility computations.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE
CERTIFICATE OF CORRECTION

PATENT NO. : 6,618,047 B1
DATED : September 9, 2003
INVENTOR(S) : Hong Lip Lim

Page 1 of 1

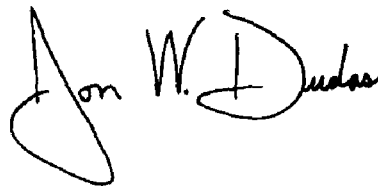
It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Column 33,

Line 56, change "claim 56" to -- claim 58 --.

Signed and Sealed this

Twentieth Day of July, 2004

A handwritten signature in black ink, reading "Jon W. Dudas". The signature is written in a cursive style with a large, stylized "J" and "D".

JON W. DUDAS

Acting Director of the United States Patent and Trademark Office

US006172679B1

(12) **United States Patent**
Lim(10) **Patent No.:** **US 6,172,679 B1**
(45) **Date of Patent:** **Jan. 9, 2001**(54) **VISIBILITY CALCULATIONS FOR 3D
COMPUTER GRAPHICS**(76) Inventor: **Hong Lip Lim**, 35 Ellie Park Avenue,
Singapore, 1545 (SG)(*) Notice: Under 35 U.S.C. 154(b), the term of this
patent shall be extended for 0 days.(21) Appl. No.: **08/935,150**(22) Filed: **Sep. 22, 1997****Related U.S. Application Data**(63) Continuation of application No. 08/182,096, filed on Jun. 1,
1994.(30) **Foreign Application Priority Data**

Jun. 28, 1991	(AU)	PK 6942
Jul. 19, 1991	(AU)	PK 7305
Oct. 1, 1991	(AU)	PK 8643
Oct. 1, 1991	(AU)	PK 8645
Oct. 30, 1991	(AU)	PK 9218

(51) **Int. Cl.⁷** **G06T 15/40**(52) **U.S. Cl.** **345/421; 345/422**(58) **Field of Search** **345/418–22, 433–9**(56) **References Cited****U.S. PATENT DOCUMENTS**

4,594,673	*	6/1986	Holly	345/421
4,625,289	*	11/1986	Rockwood	345/422
4,697,178		9/1987	Heckel	
4,819,192		4/1989	Kuragano et al.	
4,825,391	*	4/1989	Merz	345/431
4,855,938	*	8/1989	Gonzalez-Lopez et al.	345/422
4,901,252		2/1990	Fitzgerald et al.	
4,918,626	*	4/1990	Watkins et al.	345/421
4,928,250	*	5/1990	Greenberg et al.	345/426
5,027,292	*	6/1991	Matsumoto	345/422
5,058,042	*	10/1991	Hanna et al.	345/427
5,081,698		1/1992	Kohn	
5,084,830	*	1/1992	Doornink et al.	345/139 X
5,086,496		2/1992	Mulmuley	
5,088,054		2/1992	Paris, II	

5,159,663	*	10/1992	Fossum	345/422
5,253,335	*	10/1993	Mochizuki et al.	345/422
5,268,996	*	12/1993	Steiner et al.	345/426
5,295,243	*	3/1994	Robertson et al.	345/348
5,299,298	*	3/1994	Elmqvist et al.	345/421
5,313,568	*	5/1994	Wallace et al.	345/426
5,377,313		12/1994	Scheibl	
5,402,532		3/1995	Epstein	
5,414,801		5/1995	Smith	
5,448,686		9/1995	Borrel	
5,914,721	*	6/1999	Hong Lip Lim	345/421

FOREIGN PATENT DOCUMENTS

0 481 581	4/1992	(EP)
2 228 850	9/1990	(GB)

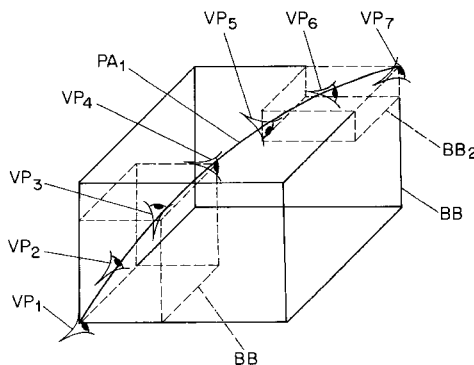
OTHER PUBLICATIONS“Stabbing Isothetic Boxes & Rectangles in $O(n \lg n)$ time”,
M. Hohmeyer S. J. Teller, 1991.*“A Characterization of Ten Hidden-Surface Algorithms”, I.
E. Sutherland, R.F. Sproull, R. A. Schumacker Computing
Surveys, vol. 6, No. 1, 1975.*

“Fast Algorithms for 3D-Graphics”, G. Glaeser, 1994.*

(List continued on next page.)

Primary Examiner—Mark R. Powell*Assistant Examiner*—Chante' Harrison(74) *Attorney, Agent, or Firm*—Oppedahl & Larson LLP(57) **ABSTRACT**

Disclosed is a method of reducing the complexity of hidden surface removal in 3D graphic systems. A fuzzy projection (FF) of a surface (SU) as seen from a number of viewpoints (VP) in a bounding box (BB) is stored in a buffer (FA) having elements (FE). A combination of all the patches (PT) of the surface (SU) viewed from a fuzzy region (FR) where surfaces can be either visible, hidden, or unable to be determined with certainty as to whether or not visible/hidden. A non-fuzzy region (NF) describes those patches (PT) that are always visible.

32 Claims, 17 Drawing Sheets

US 6,172,679 B1

Page 2

OTHER PUBLICATIONS

- "Advances in Fuzzy Sets, Possibility and Applications", 1983.*
- H. Xu Q. Peng, "Accelerated radiosity method for complex environments", Eurographics '89, 1989.*
- Application Challenges to Computational Geometry, CG Impact Task Force Report CG Impact Task Force Technical Report TR-521-96, Princeton University 1996.*
- Computer Graphics, Principles and Practice, 2nd edition J.D. Foley A. van Dam S. K. Feiner J.P. Hughes.*
- New Trends in Animation and Visualization N. Thalmann D. Thalmann 1991.*
- Graphics Systems; Architecture & Realization R. Andreov 1993.*
- Analysis of Radiosity Techniques in Computer Graphics B. Kwok MSc Thesis York University, May 1992.*
- Image display data computer forming method—uses perspective transformation with calculation time reduction on shadow and hidden surface processing Sony 86.233751/36.*
- Stabbing and ray shooting in 3 dimensional space M. Pellegrini 1990.*
- The Geometry of Beam Tracing N. Dadoun D. Kirkpatrick 1985.*
- Algorithms for line transversals in space D. Avis CG1987.
- Optimization of the binary space partition algorithm (BSP) for the visualization of dynamic scenes E. Torres Eurographics'90.*
- A mathematical Semantics of Rendering II. Approximation E. Fiume CVGIP: Graphical Models and Image Processing vol. 53, No. 1, Jan. 1991.*
- A Characterization of Ten Rasterization Techniques N. Ghurachorloo S. Gupta R. Sproull I. Sutherland Computer Graphics, vol. 23, No. 3 (Siggraph'89) 1989.*
- The use of projective geometry in Computer Graphics I. Herman 1992.*
- Ray Tracing with Cones J. Amanatides Computer Graphics, vol. 18, No. 3 (Siggraph'84) 1984.*
- The A-buffer, an Antialiased hidden surface method L. Carpenter Computer Graphics, vol. 18, No. 3, (Siggraph'84) 1984.*
- Principles and Applications of Pencil Tracing M. Shinya T. Takahashi S. Naito Computer Graphics, vol. 31, No. 4, (Siggraph'87) 1987.*
- Light-water interaction using backward beam tracing M. Wall Computer Graphics, vol. 24, No. 4 (Siggraph'90) 1990.*
- Rendering CSG Model with a ZZ-buffer D. Salcsin J. Stoll Computer Graphics, vol. 24, No. 4 (Siggraph'90) 1990.*
- A solution to the hidden-line problem for computer-drawn polyhedra P. Lourel IEEE Transactions on Computers, Mar. 1970.*
- Sorting and the hidden-surface problem I. Sutherland R. Sproull R. Schumacker National Computer Conference, 1970.*
- The Notion of quantitative invisibility and the machine rendering of solids A Appel ACM National Meeting 1967.*
- An analytic visible surface algorithm for independent pixel processing E. Catmull Computer Graphics, vol. 18, No. 3 (Siggraph'84) 1984.*
- Computing the lines piercing four lines S. Teller M. Hohmeyer Technical Report 92-665, University of California, Berkeley.*
- Computing the antipenumbra of an area light source S. Teller Computer Graphics, vol. 36, No. 2, (Siggraph'92) 1992.*
- Computer animation, theory and practice, second revised edition N. Thalmann D. Thalmann 1990, Image Sythesis M. Brest 1992.*
- Management of large amounts of data in interactive building walkthroughs T. Funkhouser C. Sequin S. Teller Symposium on Interactive 3D Graphics, 1992.*
- Temporal Coherence in Ray Tracing S. H. Badt PhD thesis, University of Texas at Dallas, 1989.*
- Near real-time shadow generation using BSP trees N. Chin S. Feiner Computer Graphics, vol. 23, No. 3 (Siggraph'89) 1989.*
- Adaptive Display algorithm for interactive frame rates during visualization of complex virtual environments T. Funkhouser C. H. Sequin Siggraph'93 1993.*
- Modeling global diffuse illumination for image synthesis A. Campbell, III. PhD Thesis University of Texas at Austin 1991.*
- A 3-dimensional representation for fast rendering of complex scenes S. Rubin T. Whitted 1980.*
- Radiosity redistribution for dynamic environment D. George F. Sillion D. Greenberg IEEE Computer Graphics & Applications, vol. 4, 1990.*
- A survey of shadow algorithms A. Woo, F. Poulin A. Fournier IEEE Computer Graphics & Applications, vol. 6, 1990.*
- Error-bounded antialiased rendering of complex environments N. Greene M. Kass Siggraph'94 1994.*
- Fast computation of shadow boundaries using spatial coherence and backprojections A. Stewart S. Ghali Siggraph'94 1994.*
- A fast shadow algorithm for area light sources using back-projection G. Drettakis E. Fiume Siggraph'94 1994.*
- Stabbing oriented convex polygons in randomized $O(n^2)$ time. S. Teller M. Hohmeyer Contemporary Mathematics, 1994.*
- Obscuration culling on parallel graphics architecture C. George Technical report TR95-017 University of North Carolina at Chapel Hill 1995.*
- Visibility between two edges of a simple polygon D. Avis T. Gum G. Goussaint The Visual Computer, 1986, No. 2.*
- Increasing update rates in the building walkthrough system with automatic model-space subdivision and potentially visible set calculations J. M. Airey PhD Thesis, University of North Carolina, 1990.*
- Realism in computer graphics: a survey. J. Amanatides IEEE Computer Graphics and Applications, vol. 7, No. 1, 1987.*
- Finding a line transversal of axis objects in three dimensions N. Amenta Proc. 3rd Annual ACM-SIAM Symposium on Discrete Algorithms, 1992.*
- Beyond the third dimension, geometry, computer graphics and higher dimension Banchoff T. F. Scientific American Library 1990.*
- A general version of crow's shadow volumes Bergeron P. IEEE Computer Graphics and Applications, vol. 6, No. 9, Sep. 1986.*
- Me and my (fake shadow) Blinn, J. F. IEEE Computer Graphics and Applications, vol. 8, No. 1 1988.*
- Generating soft shadows with a depth buffer algorithm Brotman L. S. IEEE Computer Graphics and Applications vol. 4, No. 10, 1984.*

US 6,172,679 B1

Page 3

- A multiresolution spline with application to image mosaics Burt P.J. Adelson E. H. ACM Transactions on Graphics, vol. 2, Oct. 1983.*
- Hierarchical geometric models for visible surface algorithms Clark, J. H. Communications of the ACM, vol. 19, No. 10, Oct. 1976.*
- An overview of rendering techniques Dennis A. R. Computer & Graphics, vol. 14, No. 1, 1990.*
- Hybrid shadow testing scheme for ray tracing Eo K. S. Kyung C. M. Computer Aided Design vol. 21, No. 1 Jan. 1989.*
- Hierarchical rendering of complex environments Greene, N. PhD dissertation, University of California, Santa Cruz 1995.*
- Bibliography of hidden-line and hidden-surface algorithms Griffiths, J. G. Computer-Aided Design, vol. 10, May 1978.*
- The light buffer: a shadow-testing accelerator Haines, E.A. Greenberg, D. IEEE Computer Graphics and Applications, vol. 6, No. 9, Sep. 1986.*
- Algorithms for antialiased cast shadows Houreade J. C. Nicolas A. Computer and Graphics, vol. 9, No. 3 1985.*
- Hemi-cube ray-tracing; a method for generating soft shadows Meyer U. Proceedings, Eurographics'90, 1990.*
- Principles of interactive computer graphics, 2nd edition Newman, W. M. Sproull, R. E 1979.*
- Principles of interactive computer graphics, 1st edition Newman, W. M. Sproull, R. E 1973.*
- A comparison of four visibility acceleration techniques for radiosity Ng, A. The visual computer, 1996, vol. 12, pp. 307-316.*
- Shading models for point and linear sources Nishita, T. Okamura, I. Nakamse, E. ACM transactions on graphics, vol. 4, No. 2, Apr. 1985.*
- Radiosity of dynamic scenes in flatland with the visibility complex Orti, R. Riviere S. Durand E Pucch C. Proceedings, Eurographics'96 1996.*
- Visibility, occlusion, and the aspect graph Plantinga H. Dyer C. R. International Journal of computer vision, vol. 5, No. 2, 1990.*
- Necessary and sufficient conditions for hyperplane transversal Pollack R. Wenger R. Proc. 5th Annual Symposium on Computational Geometry, 1989.*
- Shading and shadowing with linear light sources Poulin P. Amanatides J. Proceedings, Eurographics '90 1990.*
- Rendering antialiased shadows with depth maps Reeves W. Salesin D. Cook R. Proceedings, Siggraph'87 1987.*
- Machine perception of three-dimensional solids Roberts L. G. Optical and Electro-Optical Information Processing, J. Tippett (editor) MIT Press, 1965.*
- Procedural elements for computer graphics Rogers D. P. McGraw-Hill(publisher) 1985.*
- Optics B. Rossi Addison-Wesley(publisher) 1957.*
- An optimal algorithm for detecting weak visibility of a polygon J. Sack S. Suri IEEE transactions on computers, vol. 39, No. 10, Oct. 1990.*
- Shaded rendering and shadow computation for polyhedral animation Seales W. B. Dyer C. R. Proceedings, Graphics Interface '90, May 1990.*
- Optics, third edition Sears F. W. Addison-Wesley Publishing Co. 1949.*
- Linear programming and convex hulls made easy Seidol R. Proc. 6th ACM Symposium on COMputational Geometry, 1990.*
- Output-sensitive visibility algorithms for dynamic scenes with applications to virtual reality Sudarsky O. Gotsman C. Proceedings, Eurographics'96 1996.*
- Automatic view function generation for walk-through animation using a reeb graph Shinagawa Y. kunii T. Nomura Y. Okuna T. Young Y. Computer Animation '90 1990.*
- Octant priority for radiosity image rendering Wang Y. Davis W. Proceedings, Graphics Interface'90 1990.*
- Casting curved shadows on curved surfaces Williams L. Proceedings, Siggraph'78 1978.*
- Pyramidal parametrics William L., Proceedings, Siggraph'83 1983.*
- Accelerated radiosity method for complex environments Xu H. Peng Q. Liang Y. Proceedings, Eurographics'89 1989.*
- Pyramid clipping for efficient ray traversal Zwann M. Reinhard E. Jansen F. Proceedings of the Sixth Eurographics Rendering Workshop 1995.*
- Application challenges to computational geometry, CG impact task force report CG impact task force Princeton University Computer Science Dept. Technical Report TR-521-96 <http://graphics.les.mit.edu/~seth/pubs/task-force/techrep.html> <http://www.cs.princeton.edu/~chazelle/taskforce/CGreport.ps.Z> <http://graphics.les.mit.edu/~seth/pubs/pubs.html> 1996.*
- My response to application challenges to computational geometry Franklin R. http://www.ecse.rpi.edu/Homeworks/wrf/geom_response.html http://netlib.boll-labs.com/netlib/compgeom/discuss/archive/96/ta_skforce.html 1996.*
- Comments on the report "Application challenges to computational geometry" Heckbert P. <http://www.cs.duke.edu/~jette/compgeom/files/heckbert.html> http://netlib.bell-labs.com/netlib/compgeom/discuss/archive/96/ta_skforce.html 1996.*
- Follow-up comments on the report "Application challenges to computational geometry" Coulson T. <http://www.cs.duke.edu/~jette/compgeom/files/coulson.html> http://netlib.bell-labs.com/netlib/compgeom/discuss/archive/96/ta_skforce.html 1996.*
- Inside Quake: visible surface determination <http://www.gamers.org/dEngine/quake/papers/ddjpv.html> 1996.*
- CGDC Quake Talk <http://www.gamers.org/dEngine/quake/papers/mikeab-egde.html> 1996.*
- Quake hidden surface removal <http://www.gamers.org/dEngine/quake/papers/ddjzsort.html> 1996.*
- Quake editing tools information http://www.gamers.org/dEngine/quake/QuakeEd/qedit_infor.html 1996.*
- Zen of graphics programming Abrash M. 1996.*
- Computer graphics: more unsolved problems Siggraph'91 panel 1991.*
- Global illumination in architecture and entertainment Siggraph'96 course notes 1996.*
- Interactive walkthrough of large geometric database Siggraph'96 course notes 1996.*
- Imprecise computation and load sharing in computer generated imaging system Berger M. Zhao W. Graphics Interface '90 1990.*
- Exploiting temporal coherence in ray tracing Chapman J. Calvert T. Sill J. Graphics Interface'90 1990.*
- Approximate ray tracing Dauenhauer D. Graphics Interface '90 1990.*
- Approximate and probabilistic algorithms for shading and rendering structured particle systems Reeves W. Siggraph 1985 1985.*

US 6,172,679 B1

Page 4

- Fundamentals of interactive computer graphics Foley J.D. van Dam A. Addison-Wesley Publishing Co. 1982.*
- Multi-pass multi-resolution algorithm for the determination of hidden surfaces Lim, H. L. Inter-faculty symposium on computer graphics and image processing 1987.*
- Ten Unsolved Problems in Rendering Heckbert P. S. Workshop on Rendering Algorithms and Systems, Graphics Interface '87 www addr: <http://www.cs.cmu/~ph> (index page) 1987.*
- 3D comes alive Ozer, J. PC magazine, Jun. 25, 1996.*
- Affordable 3-D workstations Hummel, R. L. Byte Dec. 1996.*
- Gaming: in the next dimension Case, L. Salvator, D. Computer Gaming Jul. 1996.*
- 3D engine list Isakovic, K. www addr: <http://www.cs.lum-berline.de/~ki/engines.html> 1996.*
- OpenGL the leading visual programming interface www addr: http://www.agi.com/Products/Dev_environs_ds.html 1996.*
- 3D computer graphics (second edition) Glassner A. S. Design Press 1989.*
- The UC Berkeley system for interactive visualization of large architectural models T. Funkhouser S. Teller C. Sequin D. Khorramabadi Presence, vol. 5, No. 1, Winter 1996.*
- Visibility computation for efficient walkthrough of complex environment R. Yagel R. William Presence, vol. 5, No. 1, Winter 1996.*
- Large models for virtual environments; A review of work by the architectural walkthrough project at UNCM, R. Mine H. Weber Presence, vol. 5, No. 1, Winter 1996.*
- A hidden-line algorithm for hyperspace R. P. Burton D. R. Smith SIAM Journal of Computing vol. 11, No. 1, Feb. 1982.*
- Canonic representations for the geometries of multiple projective views Q. T. Luong T. Vieuille University of California at Berkeley CS Dept. Tech. Report UCB/CSD093-773 1993.*
- Multidimensional graphing in two-dimensional spaces T. Mihalisin E. Gawlinski J. Timlin J. Schwegler Computers in Physics, Nov./Dec. 1989.*
- A framework for global illumination in animated environments J. Mineroll J. Dorsey H. Rushmeier Rendering Techniques '95 (Proceedings of the Eurographics Workshop in Dublin, Ireland Jun. 1995.*
- Shadows for bump-mapped surfaces N. L. Max Advanced Computer Graphics (Proceedings of Computer Graphics Tokyo '86) 1986.*
- The simulation of natural features using cone tracing D. Kirk Advanced Computer Graphics (Proceedings of Computer Graphics Tokyo '86) 1986.*
- Simulating soft shadows with graphics hardware P. S. Heckbert M. Hert Carnegie-Melon University CS Dept. Technical Report CMU-CS-97-104 1997.*
- Spatial transformation for rapid scan-line surface shadowing P. Robertson IEEE computer graphics & applications Mar. 1989.*
- Incremental update of the visibility map as seen by a moving viewpoint in two dimensions S. Ghali A. J. Steward Eurographics Workshop on Animation and Simulation, Aug. 1996.*
- Z. H. Zhao D. Dobkin Continuous algorithms for visibility: the space searching approach fourth eurographics workshop on rendering 1993.*
- Y. Shinagawa S. Miyoshi T. Kunii Viewpoint analysis of drawings and paintings rendered using multiple viewpoints: cases containing rectangular objects Fourth Eurographics Workshop on Rendering 1993.*
- E Jansen A. Chalmers Realism in real time Fourth Eurographics Workshop on Rendering 1993.*
- E. Catmull, "Computer display of curved surfaces" in IEEE Transactions of Computers, p. 309-315, 1971.*
- M.F. Cohen et al., "The hemi-cube: A Radiosity solution for complex environment", Computer Graphics (Siggraph '85), vol. 19, No. 3, p. 31-40, 1985.*
- G.A. Crocker, "Invisibility coherence for faster scan-line hidden surface algorithms", Computer Graphics, vol. 18, No. 3, p. 95-102, 1984.*
- H. Hubschman et al., "Frame-to-frame coherence and the hidden surface computations: Constraints for a Convex World", Computer Graphics, vol. 15, No. 3, p. 45-54, 1981.*
- H.L. Lim, "Fast hidden surface removal through structural analysis and representation of objects and their contours", Computer Graphics International '87, p. 75-88, 1987.*
- H.L. Lim, "Toward a fuzzy hidden surface algorithm", Computer Graphics International '92, p. 621-635, 1992.*
- H.L. Lim, "An efficient hidden surface algorithm for polyhedral surfaces" in International Conference on Computer & Communications in Science & Technology, Beijing, China, 1986.*
- C. Hornung, "A method for solving the visibility problem", IEEE Computer Graphics & Applications, vol. 4, p. 26-33, 1984.*
- J. Griffiths, "A depth-coherence scanline algorithm for displaying curved surfaces", Computer-aided Design, vol. 16, No. 2, p. 91-101, 1984.*
- E.A. Haines et al., "Shaft culling for efficient ray-traced radiosity", Proceedings of Eurographics Workshop on Rendering, Jul., 1991.*
- H. Plantinga et al., "Real-time hidden-line elimination for a rotating polyhedral scene using the aspect representation", in Graphics Interface '90 Proceedings, p. 9-16, 1990.*
- J.M. Airey et al., "Towards image realism with interactive update rates in complex virtual building environment", in ACM Siggraph Special Issue on the 1990 Symposium on Interactive 3D Graphics vol. 24, p. 41-50, 1990.*
- Y. Wang, "Image synthesis using front-to-back based radiosity methods", Ph.D. thesis, University of Alberta, 1992.*
- D.R. Baum et al., "The back-buffer algorithm: an extension of the radiosity method to dynamic environments", The Visual Computer, vol. 2, No. 5, p. 298-306, 1986.*
- S.J. Teller et al., "Visibility preprocessing for interactive walkthroughs", Computer Graphics, vol. 25, No. 4, p. 61-69, 1991.*
- J. Marks et al., "Image and intervisibility coherence in rendering", Graphics Interface '90 Proceedings, p. 17-30, 1990.*
- H.L. Lim, "Rendering techniques in three-dimensional computer graphics", Ph.D. thesis, University of Sydney, 1993.*
- S.J. Teller, "Visibility computations in densely occluded polyhedral environment", Ph.D. thesis, University of California at Berkeley, 1992.*
- A.S. Glassner, "Principles of Digital Image Synthesis", vol. 2, Morgan Kaufmann Publisher, San Francisco, 1995.*

US 6,172,679 B1

Page 5

- D. Gordon et al., "Front-to-back display of BSP trees", *IEEE Computer Graphics & Applications*, vol. 11, p. 79-85, 1991.*
- K.L. Shelley et al., "Path specification and path coherence", *Computer Graphics*, vol. 16, No. 3, p. 157-161, 1982.*
- J. Vilaplana et al., "Exploiting coherence for clipping and view transformations in radiosity algorithms", in *Eurographics Workshop on Photosimulation, Realism and Physics in Computer Graphics*, Rennes, France, p. 137-149, 1989.*
- C.B. Jones, "A new approach to the 'hidden line' problem", *Computer Journal*, vol. 14, No. 3, p. 232-236, 1971.*
- F.C. Crow, "Shadow algorithms for computer graphics", *Computer Graphics*, vol. 11, No. 2, p. 442-448, 1977.*
- X. Pueyo, "The use of visibility coherence for radiosity computation", in *First International Conference on Visualization and Intelligent Design in Engineering and Architecture*, p. 17-28, 1993.*
- T. Nishita et al., "Continuous tone representation of three-dimensional objects taking account of shadows and interreflection", *Computer Graphics*, vol. 19, No. 3, p. 23-30, 1985.*
- T. Funkhouser, "Database and display algorithms for interactive visualization of architectural models", Ph.D. thesis, University of California at Berkeley, 1993.*
- S.J. Teller et al., "Global visibility algorithms for illumination computations", in *Computer Graphics (Siggraph '93)*, vol. 27, p. 239-46, 1993.*
- S. Coorg et al., "Temporally coherent conservative visibility", in *Twelfth Annual ACM Symposium on Computational Geometry*, Philadelphia, ACM Press, New York, p. 1-10, 1996.*
- S. Coorg et al., "A spatially and temporally coherent object space visibility algorithm", *Laboratory of Computer Science, MIT, Technical Report TM-546*, 1996.*
- D.P. Luebke et al., "Portal and mirrors: simple, fast evaluation of potentially visible sets", in *Proceedings 1995 Symposium on Interactive 3-D Graphics*, ACM Press, New York, 1995.*
- H. Plantinga, "Conservative visibility preprocessing for efficient walkthroughs of 3D scenes", *Graphics Interface '93 Proceedings*, p. 166-173, 1993.*
- E. Catmull, "A subdivision algorithm for computer display of curved surfaces", Ph.D. thesis, Utah University, Dec. 1974.*
- J. Arvo et al., "A survey of ray tracing acceleration techniques" in *An Introduction to Ray Tracing*, editor: A.S. Glassner, Academic Press, London, p. 201-262, 1989.*
- A. Watt et al., "Advanced Animation and Rendering Techniques, Theory and Practice", ACM Press, New York, 1992.*
- F. Sillion et al., "Radiosity and Global Illumination", Morgan Kaufmann Publisher, San Francisco, 1994.*
- D.R. Baum et al., "Improving radiosity solutions through the use of analytically determined form-factors", *Computer Graphics*, vol. 23, No. 3, p. 325-334, 1989.*
- A. Fournier et al., "On the power of the frame buffer", *ACM Transaction on Graphics*, vol. 7, No. 2, 103-128, 1988.*
- C.W. Grant, "Integrated analytic spatial & temporal anti-aliasing for polyhedra in 4-Space", *Computer Graphics*, vol. 19, No. 3, p. 79-84, 1985.*
- P. Hsiung et al., "T-Buffer: fast visualization of relativistic effects in spacetime", *ACM Siggraph Special Issue on the 1990 Symposium on Interactive 3D Graphics* 24, p. 83-88, 1990.*
- A. Inselberg, "The Plane with parallel coordinates", *The Visual Computer*, vol. 1, p. 69-91, 1985.*
- N.L. Max et al., "A two-and-a-half-D motion-blur algorithm", *Computer Graphics (Siggraph '85)*, vol. 19, No. 3, p. 85-93, 1985.*
- K.V. Steiner et al., "Hidden volumes: the 4th dimension", *Computer Graphics World*, p. 71-74, Feb. 1987.*
- L.A. Zadeh, "Fuzzy Sets", *Information and Control*, vol. 8, p. 338-353, 1965.*
- W. Siedleckiet al., "Mapping techniques for exploratory pattern analysis" in *Pattern Recognition and Artificial Intelligence*, E.S. Gelsema, L.N. Kanal (eds), Elsevier, New York, p. 277-299, 1988.*
- S. Chang, "On fuzzy mapping and control", *IEEE Transactions on Systems, Man & Cybernetics*, vol. SMC-2, No. 1, p. 30-34, 1972.*
- R. Jain et al., "Imprecision in computer vision" in *Advances in Fuzzy Sets, Possibility and Applications*, P. Wang, (ed), Plenum Press, New York, p. 217-236, 1983.*
- N. Greene et al., "Hierarchical z-buffer visibility", *Computer Graphics (Siggraph '93)*, vol. 27, p. 231-238, 1993.*
- D. Greenberg et al., "Radiosity: a method for computing global illumination", *The Visual Computer*, vol. 2, p. 291-297, 1986.*
- H. Fuchs et al., "New real-time shaded display of rigid objects", *Computer Graphics*, vol. 17, No. 3, p. 65-72, 1983.*
- C.M. Hoffman et al., "Some techniques for visualizing surfaces in four-dimensional space", *Computer-aided Design*, vol. 23, No. 1, p. 83-91, 1991.*
- J.M. Lane et al., "Scan line methods for displaying parametrically defined surfaces", *Communications of the ACM*, vol. 23, No. 1, p. 23-34, 1980.*
- J.A. Gualtieri et al., "The visual potential: one convex polygon", *Computer Vision, Graphics and Image Processing*, vol. 46, No. 1, p. 96-130, 1989.*
- Z. Gigus et al., "Computing the aspect graph for line drawings of polyhedral objects", in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Computer Society Press, New York, p. 654-661, 1988.*
- P.S. Heckbert et al., "Beam tracing polygonal objects", *Computer Graphics*, vol. 18, No. 3, p. 119-127, 1984.*
- J. Zhou, "Visualization of four dimensional space and its applications", Ph.D. thesis, Purdue University, 1991.*
- R. Jain, "Application of fuzzy sets for the analysis of complex scenes" in *Advances in Fuzzy Set Theory and Applications*, M.M. Gupta et al. (eds), North-Holland, Amsterdam, p. 577-583, 1979.*
- D. Tost et al., "A definition of frame-to-frame coherence", *Computer Animation '90*, p. 207-221, 1990.*
- Y. Leung, "Spatial Analysis and Planning under Imprecision", North Holland, Amsterdam, 1988.*
- J.R. Wallace et al., "A ray tracing algorithm for progressive radiosity", *Computer Graphics*, vol. 23, No. 3, p. 315-324, 1989.*
- H. Fuchs et al., "On visible surface generation by a priori tree structure", *Computer Graphics (Siggraph'80)*, vol. 14, p. 124-133, 1980.*
- M.F. Cohn et al., "A progressive refinement approach to fast radiosity image generation", *Computer Graphics*, vol. 22, No. 4, p. 75-84, 1988.*
- K. Kanatani, "Group-theoretical Methods in Image Understanding", Springer Verlag, Berlin, 1990.*

US 6,172,679 B1

Page 6

- J.K. Aggarwal et al., "Dynamic scene analysis" in *Image Sequence Processing and Dynamic Scene Analysis*, T.S. Huang (ed), Springer Verlag, Berlin, p. 40–73, 1983.*
- N.I. Badler et al., "Motion: Representation and Perception" Elsevier, New York, 1986.*
- Subbarao, "Interpretation of visual motion: A computational study", Pitman, London, 1988.*
- F. Sillion et al., "A general two-pass method integrating specular and diffuse reflection", *Computer Graphics*, vol. 23, No. 3, p. 335–344, 1989.*
- R.J. Recker et al., "Acceleration Techniques for Progressive Refinement Radiosity", *ACM Siggraph Special Issue on the 1990 Symposium on Interactive 3D Graphics*, vol. 24, p. 59–66, 1990.*
- E. H. Ruspini, "A new approach to clustering", *Information and Control* 15, p. 22–32, 1969.*
- A. Kaufmann, "Theory of Fuzzy Subsets. Vol. I, Fundamental Theoretical Elements", Academic Press, London, 1975.*
- T.S. Huang, "Image Sequence Processing", Springer Verlag, Berlin, 1981.*
- P.A. Ligomenides, "Modeling uncertainty in human perception" in *Uncertainty in Knowledge-Based Systems*, B. Bouchon, R. Yager (eds), Springer Verlag, Berlin, p. 337–346, 1986.*
- A. Inselberg, "N-Dimensional Graphics. Part I. Lines & Hyperplanes", IBM Scientific Center Report G320–2711, Jul. 1981.*
- A.S. Glassner, "3D Computer Graphics: A User's Guide for Artists & Designers", 2nd edition, Design Press, New York, p. 139–158, 1989.*
- M.F. Cohen et al., "Radiosity & realistic image synthesis", Academic, New York, 1993.*
- J. Aggarwal et al., "Analysing dynamic scenes containing multiple moving objects" in *Image Sequence Analysis*, editor: T.S. Huang, Springer-Verlag, Berlin, p. 355–380, 1981.*
- A. Appel, "The Notion of quantitative invisibility and the machine rendering of solids", *Proceedings ACM National Conference*, Thompson Books, Washington, DC, p. 387–393, 1967.*
- J. Vince, "Computer Animation", Addison-Wesley, New York, 1992.*
- Y. Chrysanthou et al., "Computing dynamic changes to BSP-trees", *Eurographics '92*, vol. 11, No. 3, p. C–321–C–332, 1992.*
- S. Ansoldi et al., "Geometric modeling of solid objects by using a face adjacency graph presentation" *Computer Graphics (Siggraph '85)*, vol. 19, No. 3, p. 131–138, 1985.*
- N. Chin et al., "Fast object-precision shadow generation for area light sources using BSP trees", *Proceedings 1992 Symposium on Interactive 3D Graphics*, p. 21–30, 1992.*
- D.S. Immel et al., "A radiosity method for non-diffuse environments", *Computer Graphics*, vol. 4, p. 133–142, 1986.*

* cited by examiner

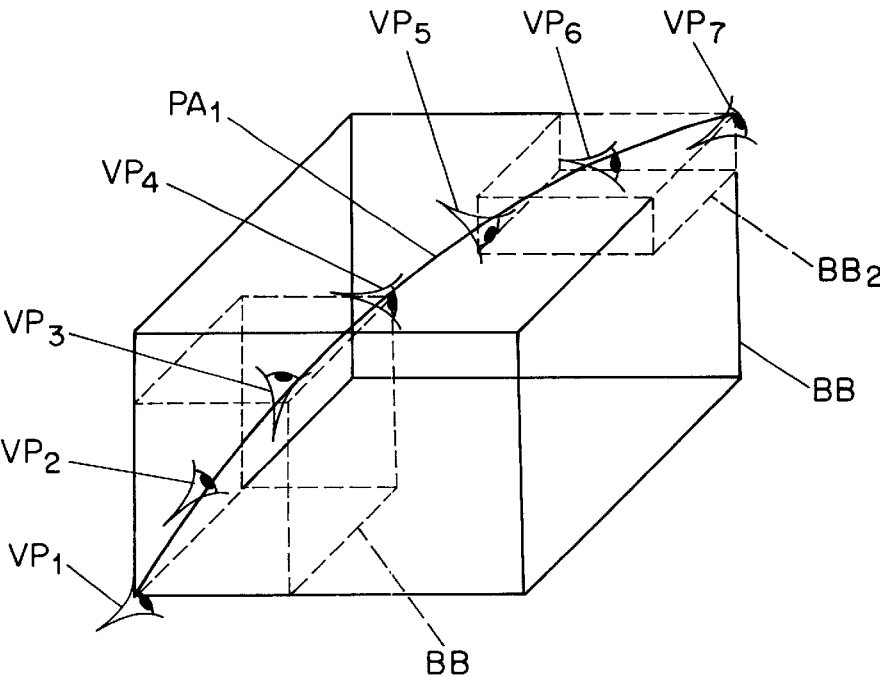


FIG. 1A

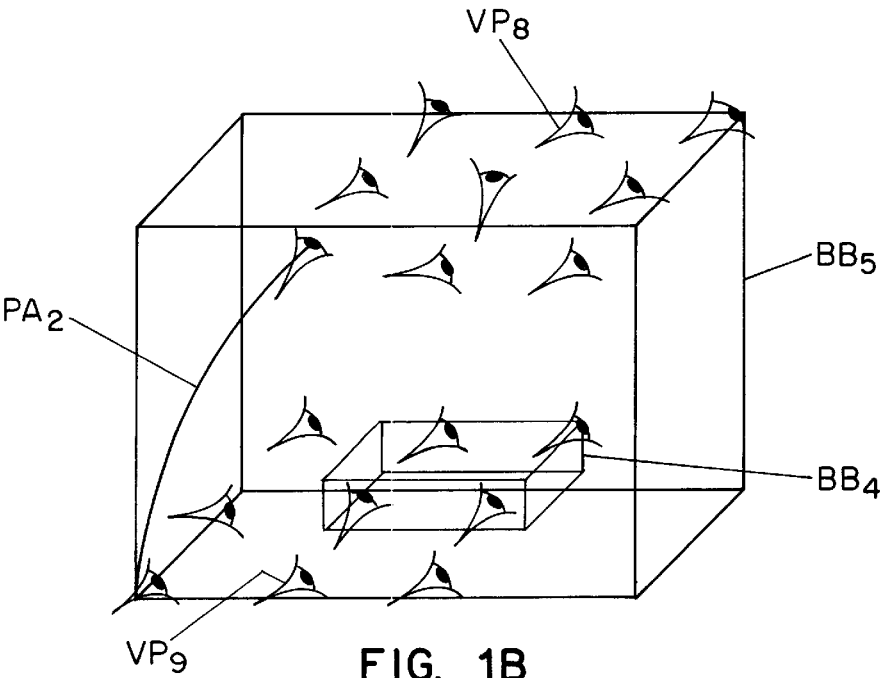


FIG. 1B

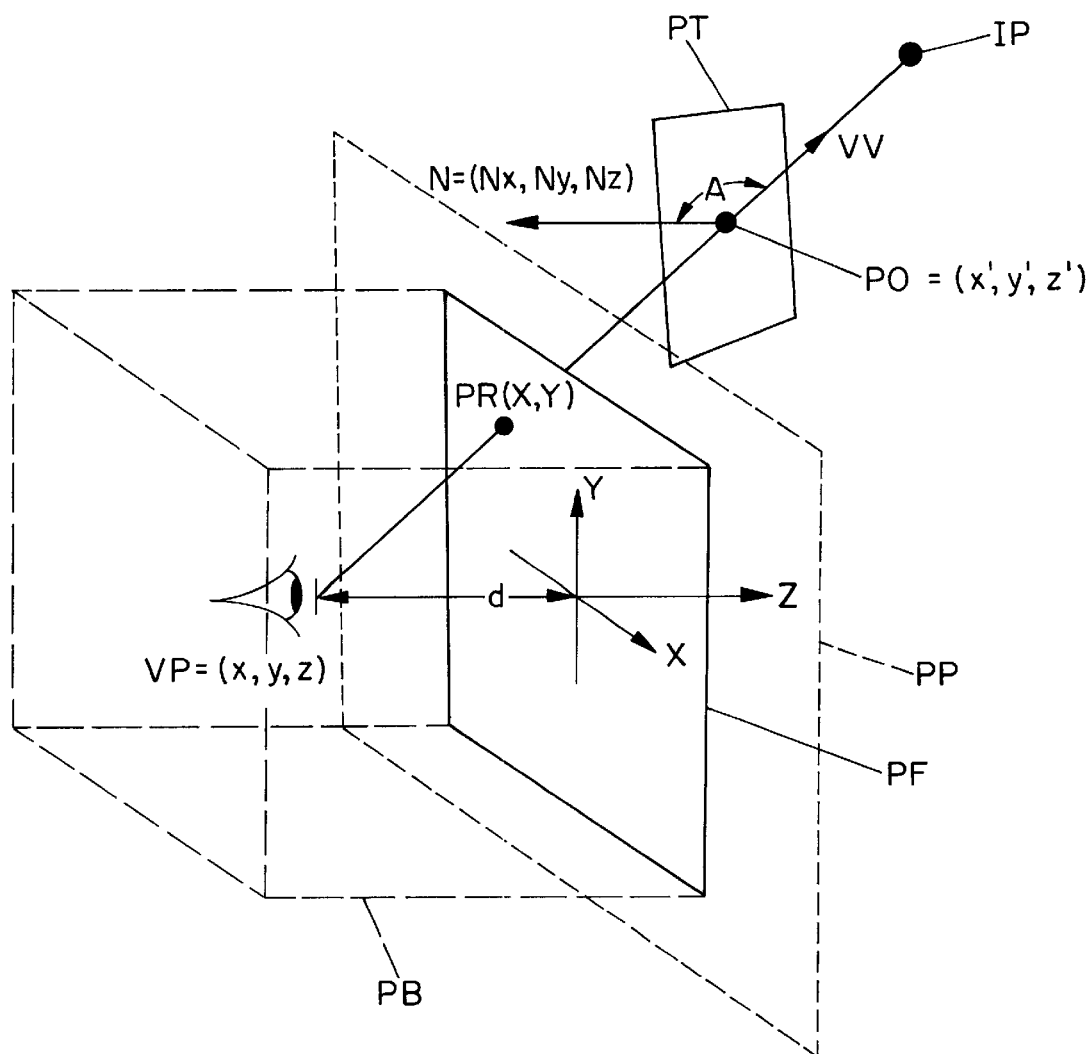


FIG. 2

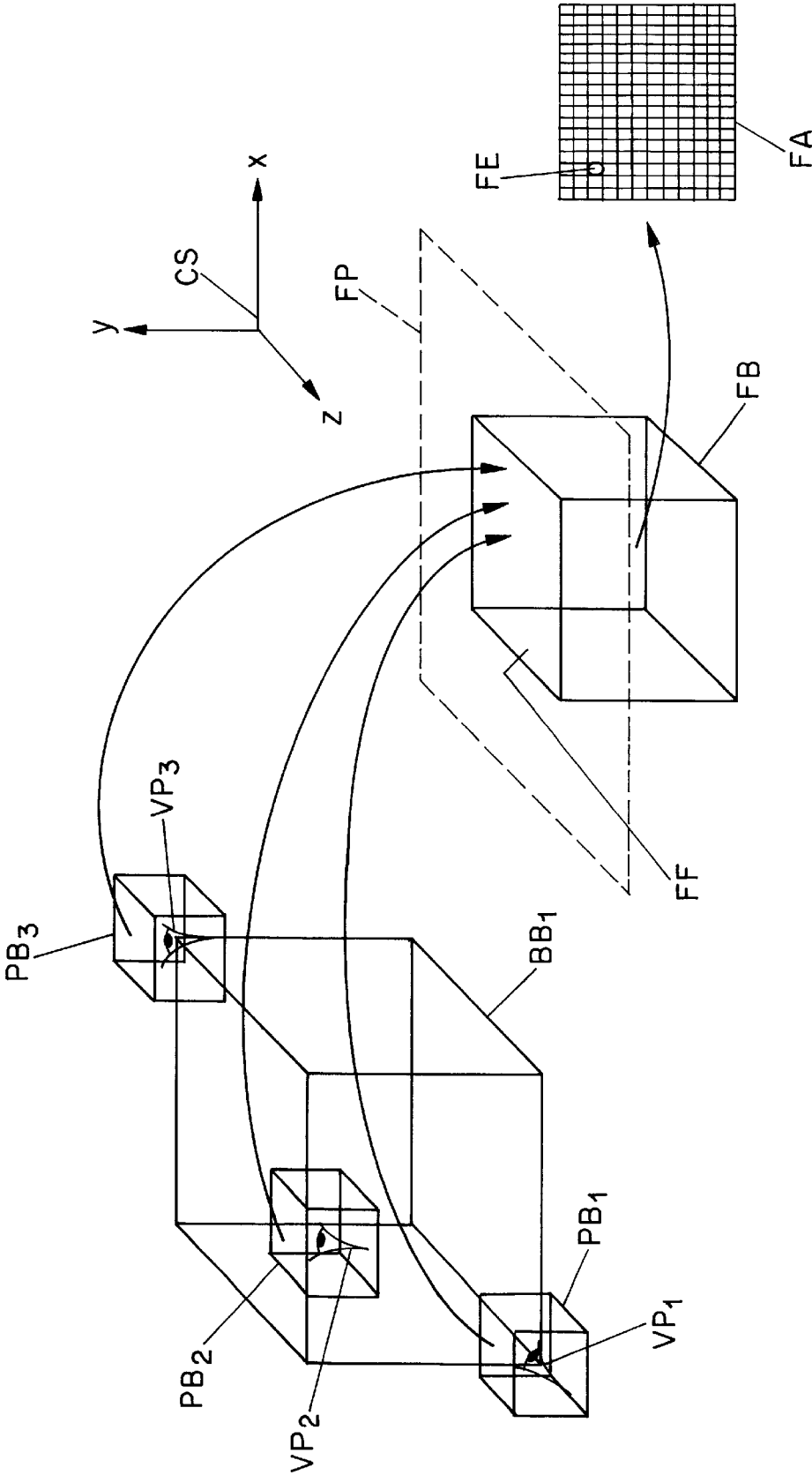


FIG. 3

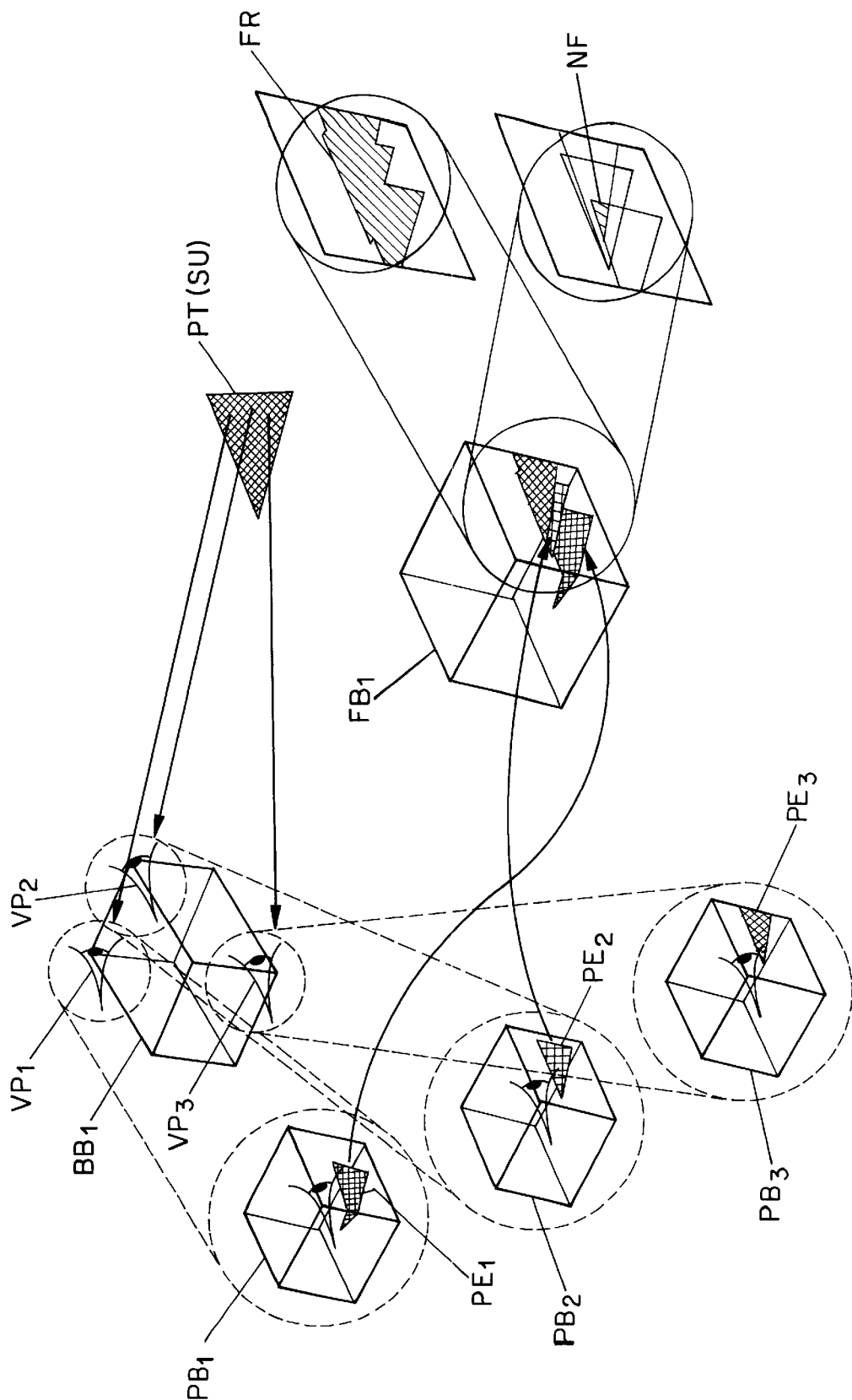


FIG. 4

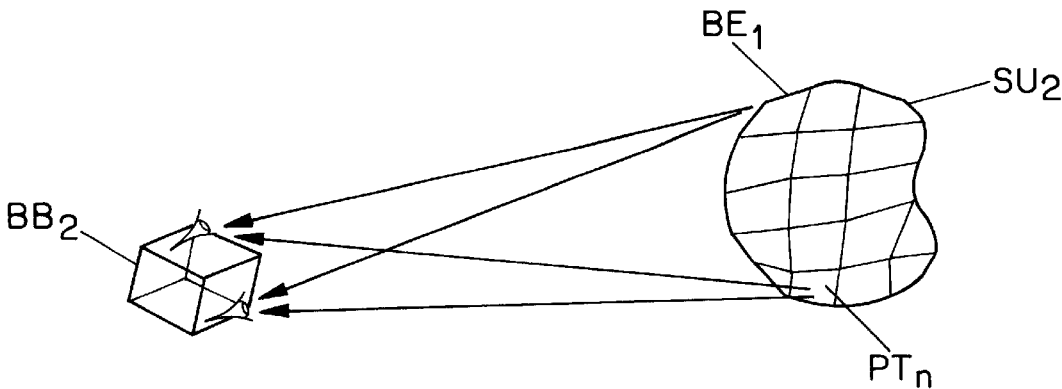
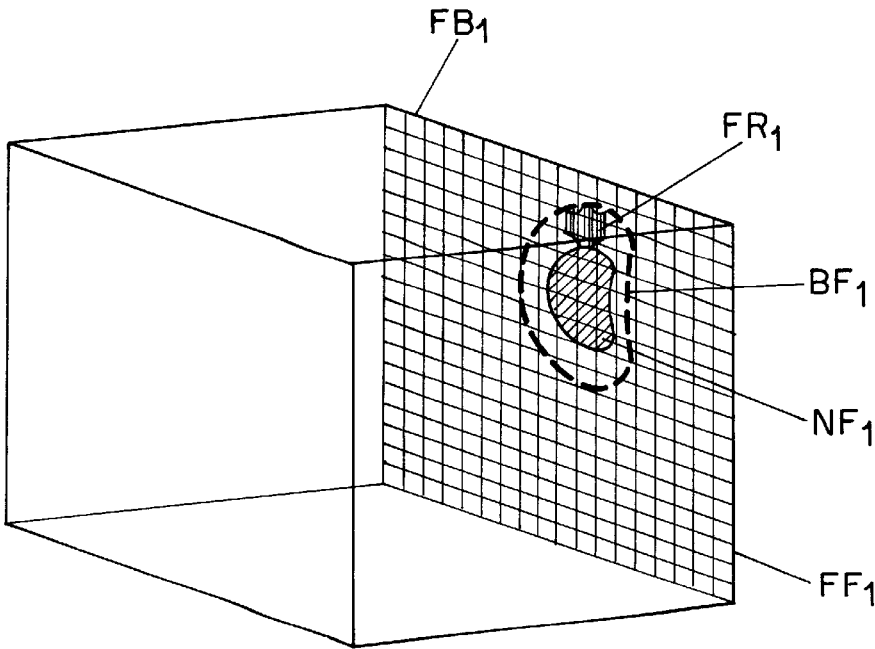


FIG. 5A



LEGEND:




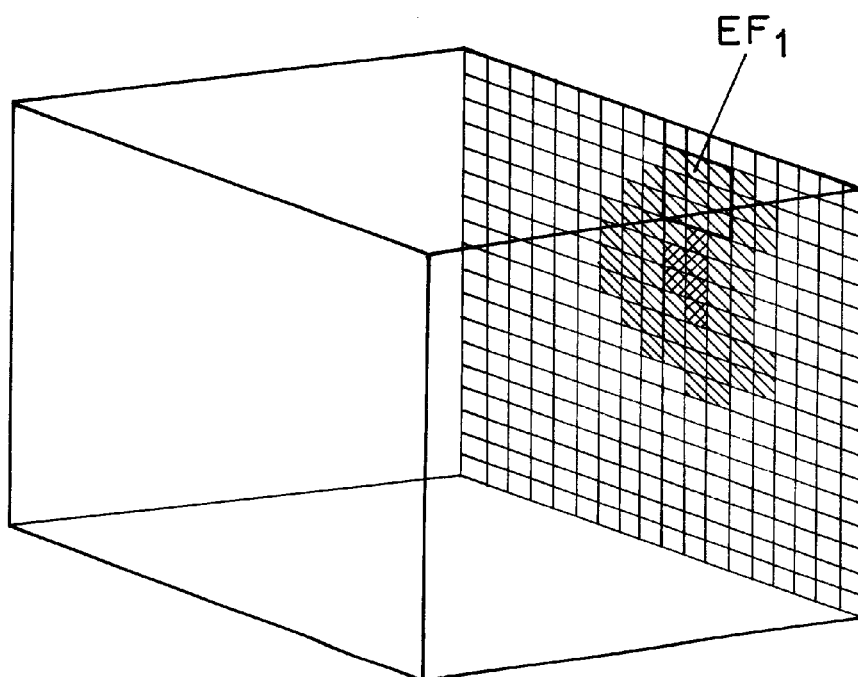
-  BORDER OF THE FUZZY REGION
-  THE FUZZY REGION OF A BOUNDARY EDGE
-  NON-FUZZY REGION

FIG. 5B



LEGEND:



REGION COMBINING THE FUZZY
EXTENT OF BOUNDARY EDGES



APPROXIMATED NON-FUZZY REGION
COMPUTED BY SUBTRACTION FROM THE
FUZZY EXTENT OF BOUNDARY EDGES

FIG. 5C

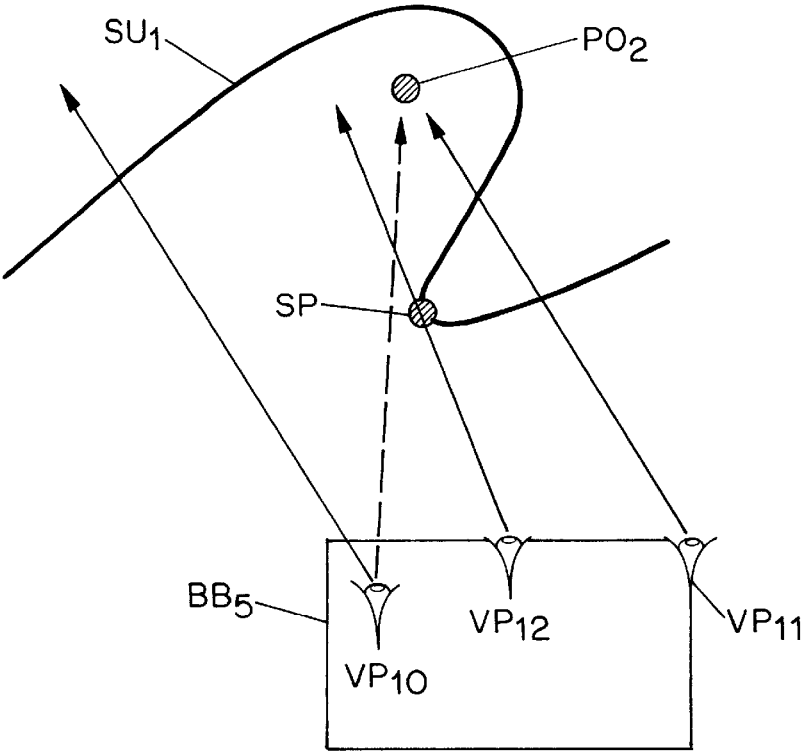


FIG. 5D

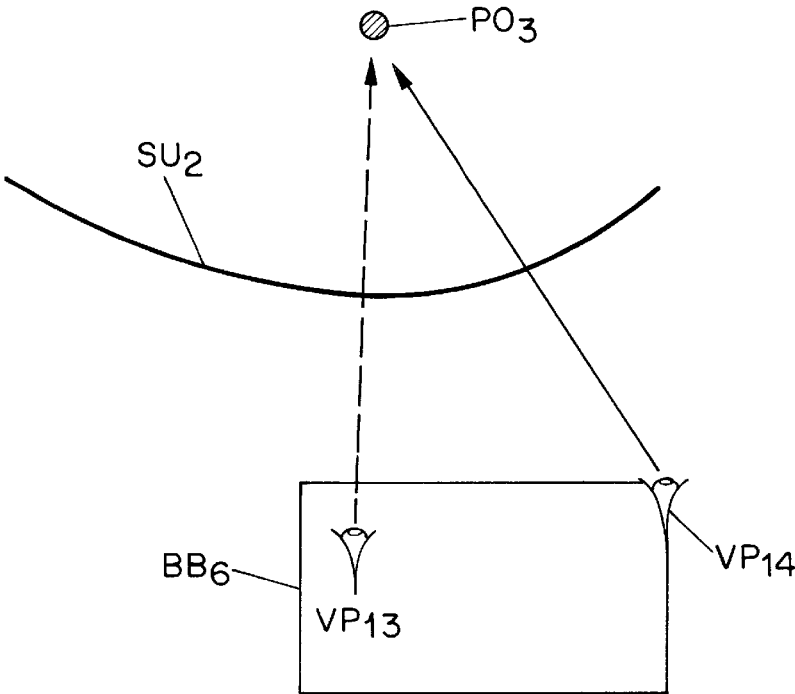
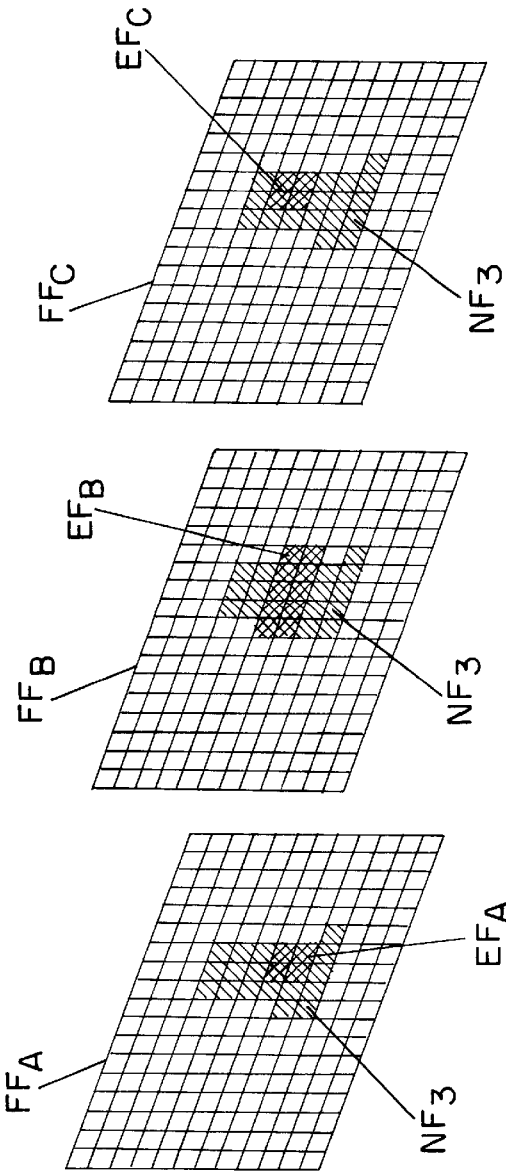
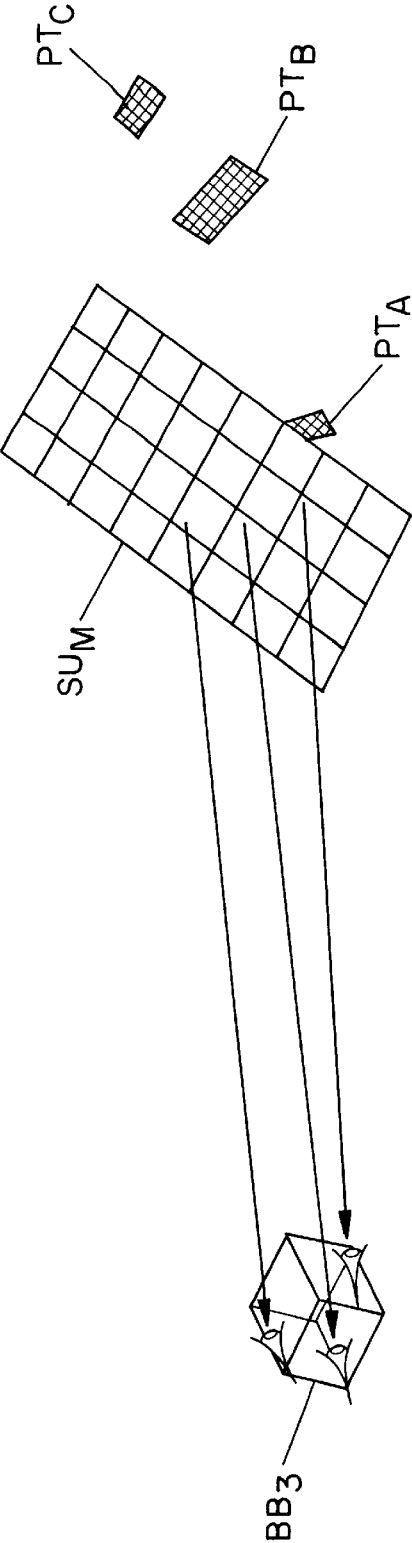


FIG. 5E



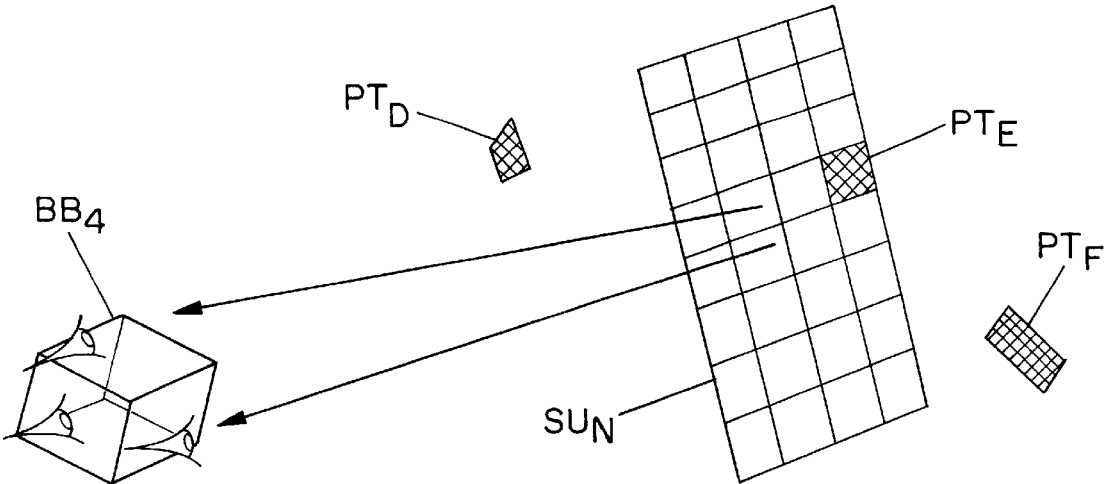


FIG. 7A

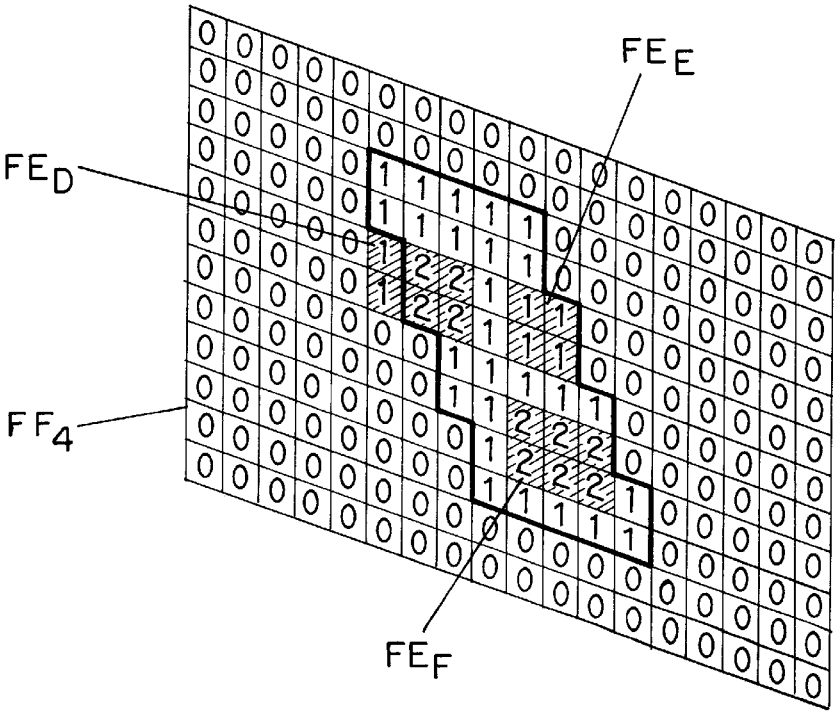


FIG. 7B

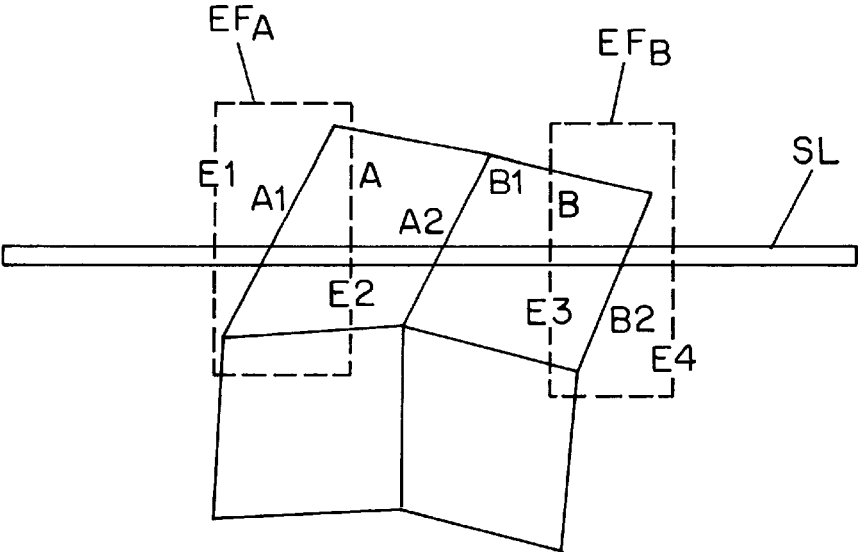


FIG. 8A

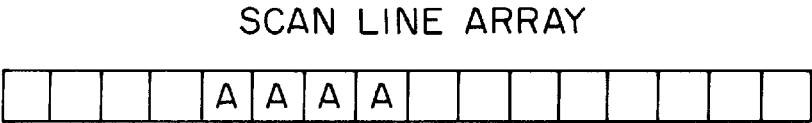


FIG. 8B

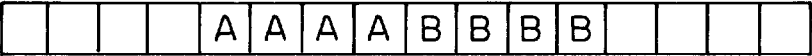


FIG. 8C



FIG. 8D

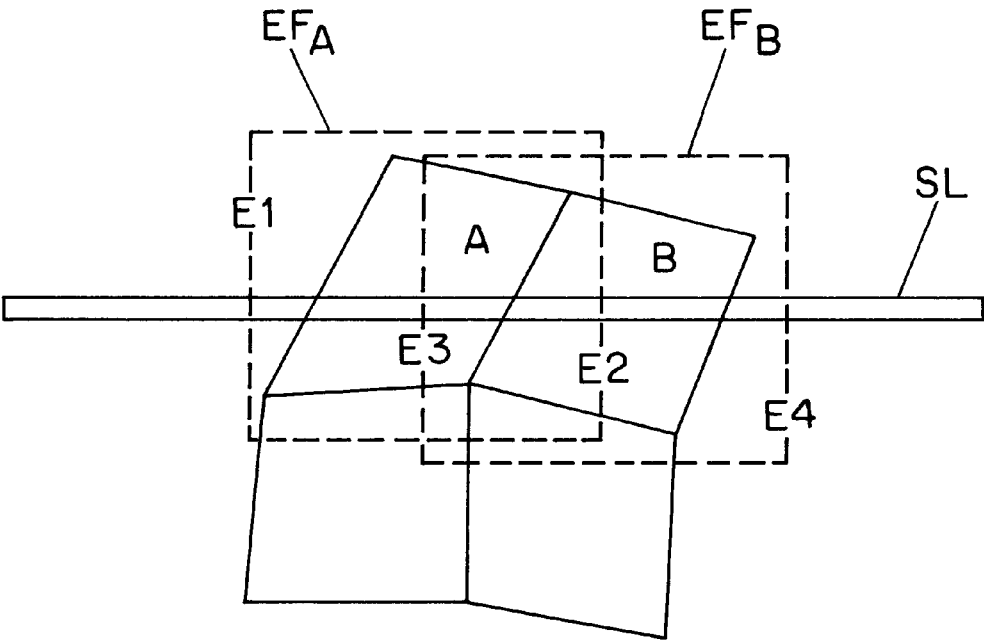


FIG. 9A

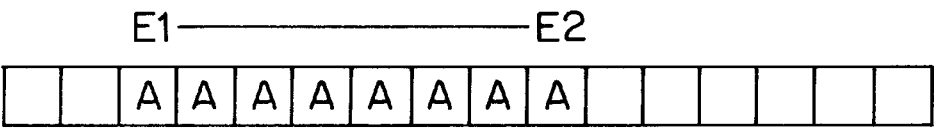


FIG. 9B

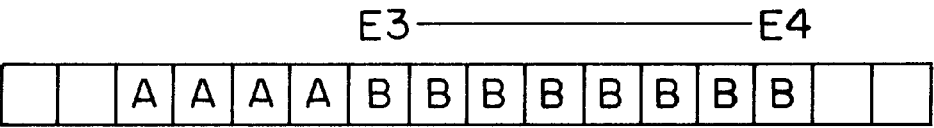
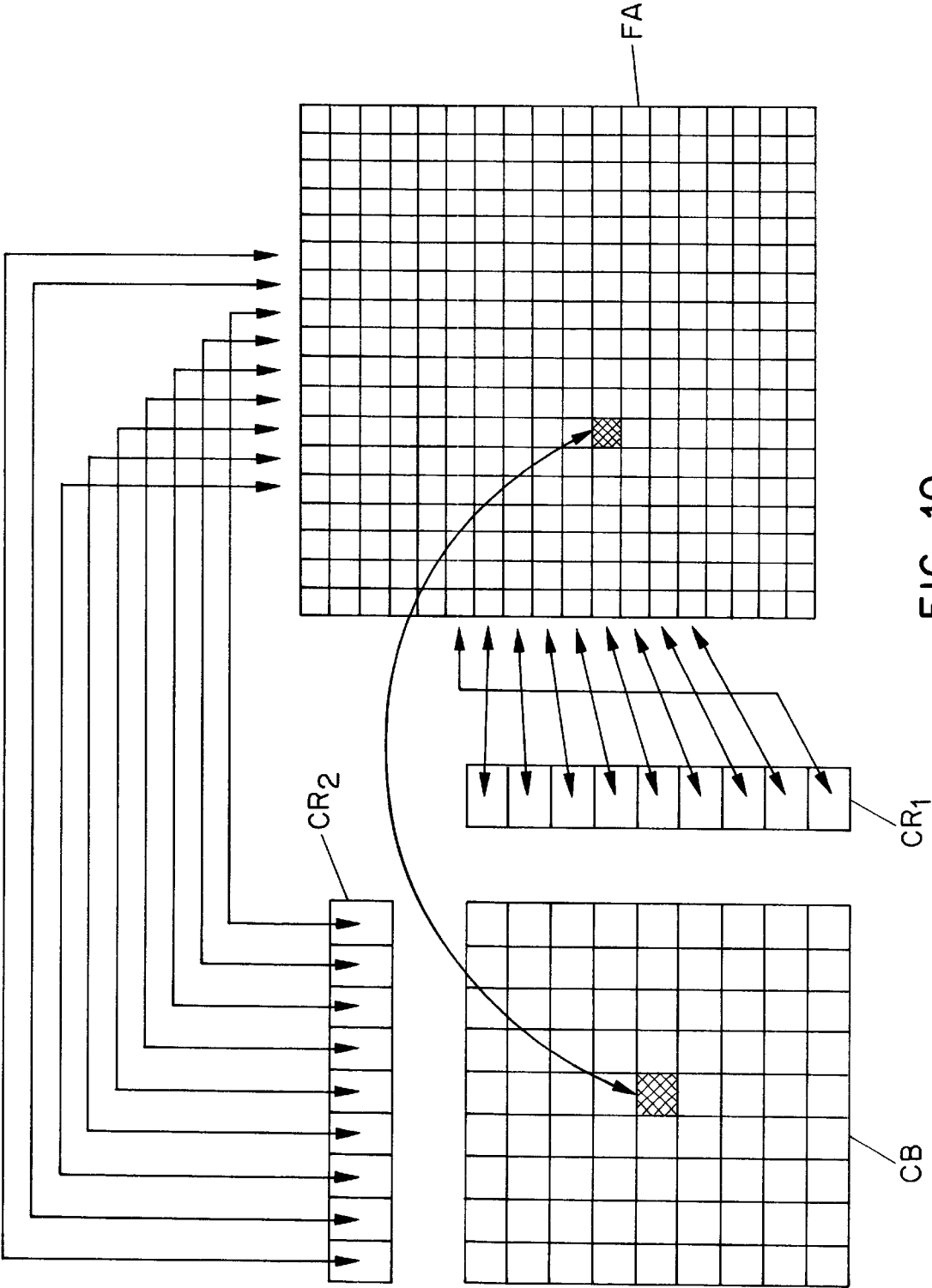


FIG. 9C



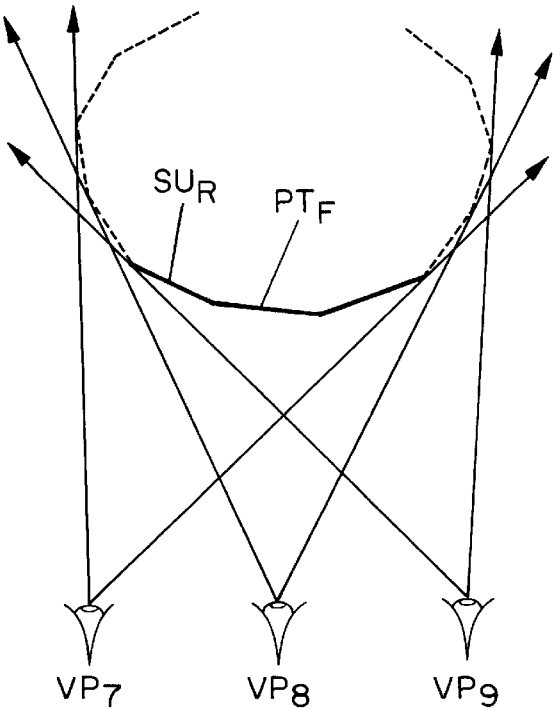


FIG. 11A

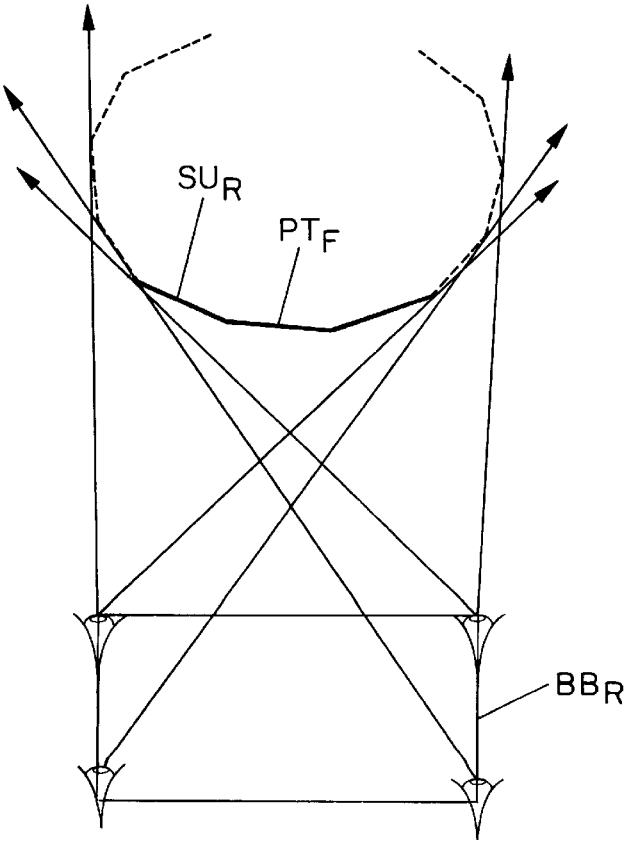


FIG. 11B

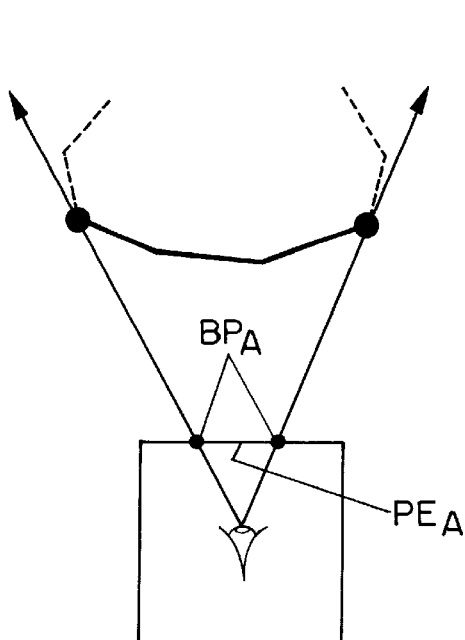


FIG. 12A

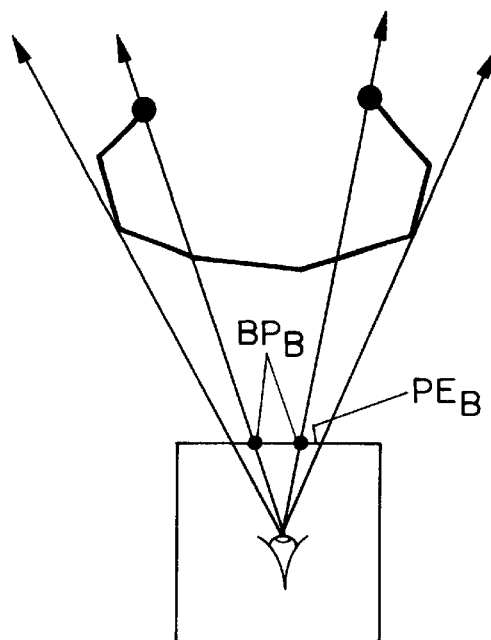


FIG. 12B

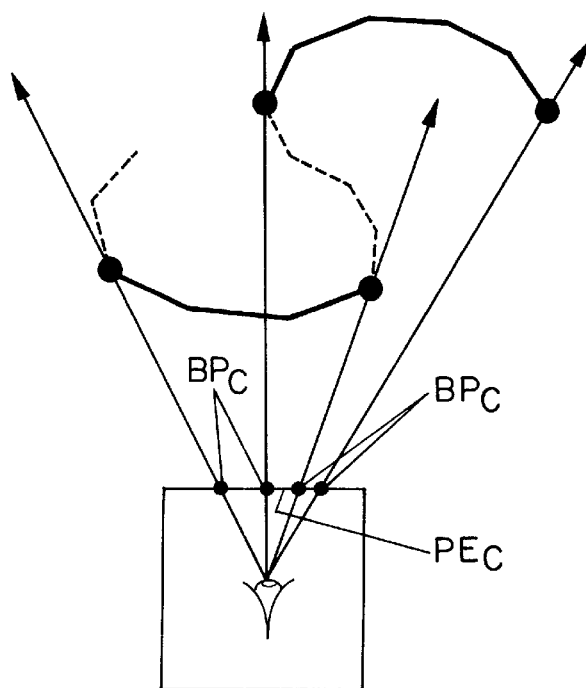


FIG. 12C

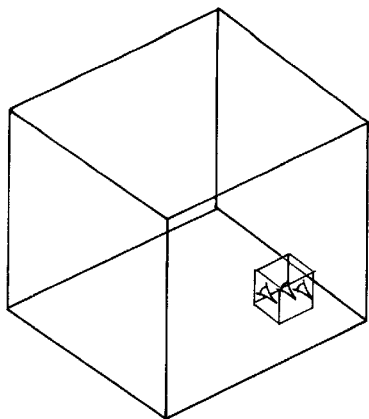


FIG. 13A

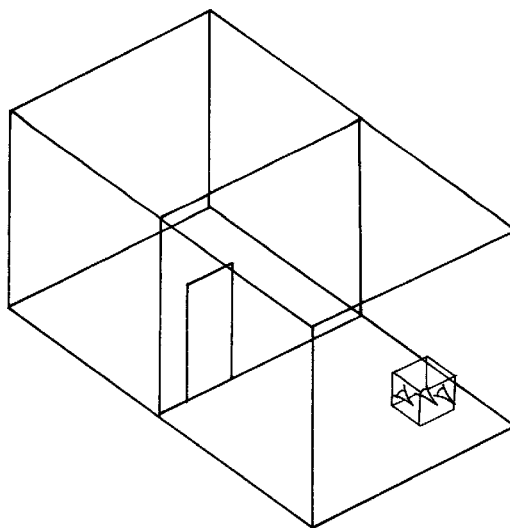


FIG. 13B

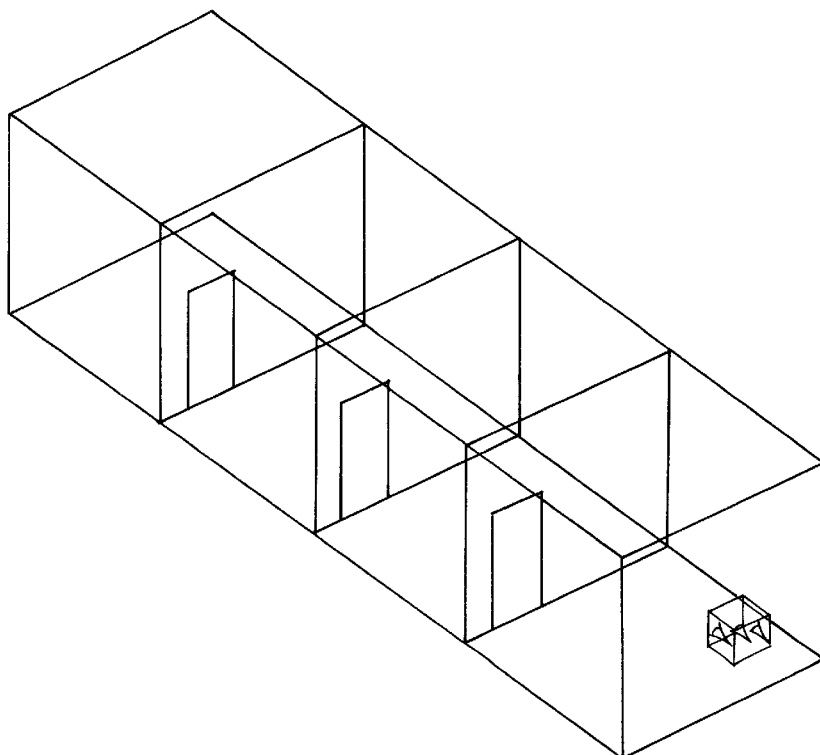
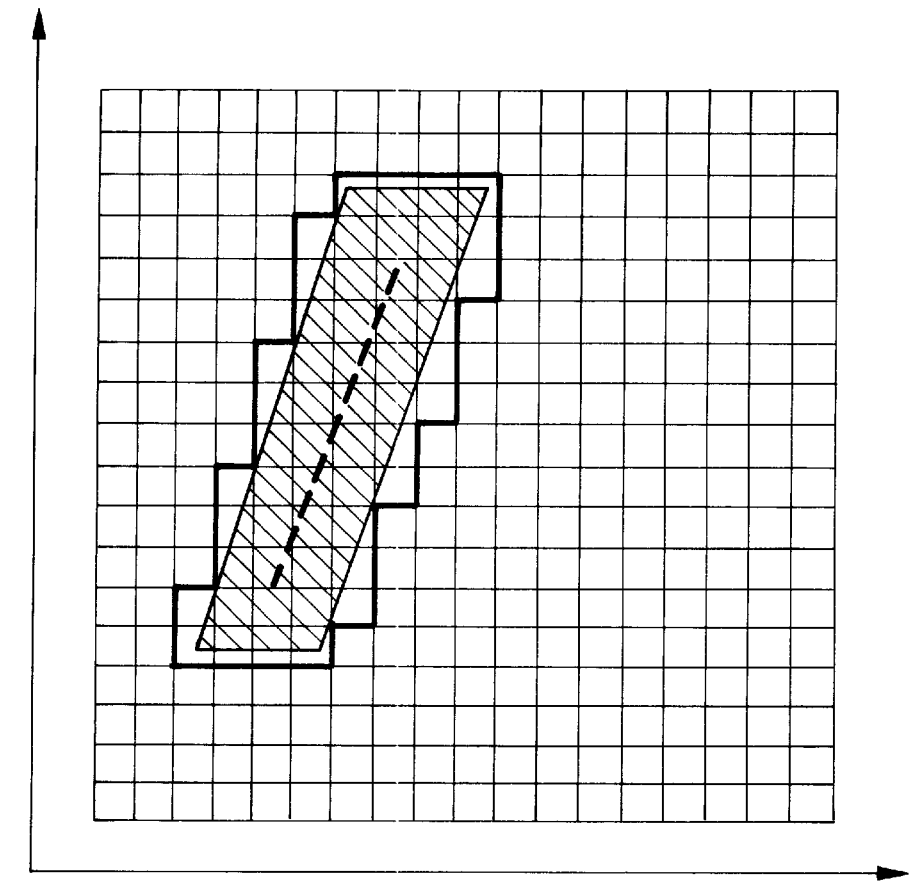


FIG. 13C



LEGEND:

----- PROJECTION OF EDGE WITH
RESPECT TO ONE VIEWPOINT

 FUZZY REGION OF EDGE

 EXPANDED FUZZY REGION
OF EDGE

FIG. 14

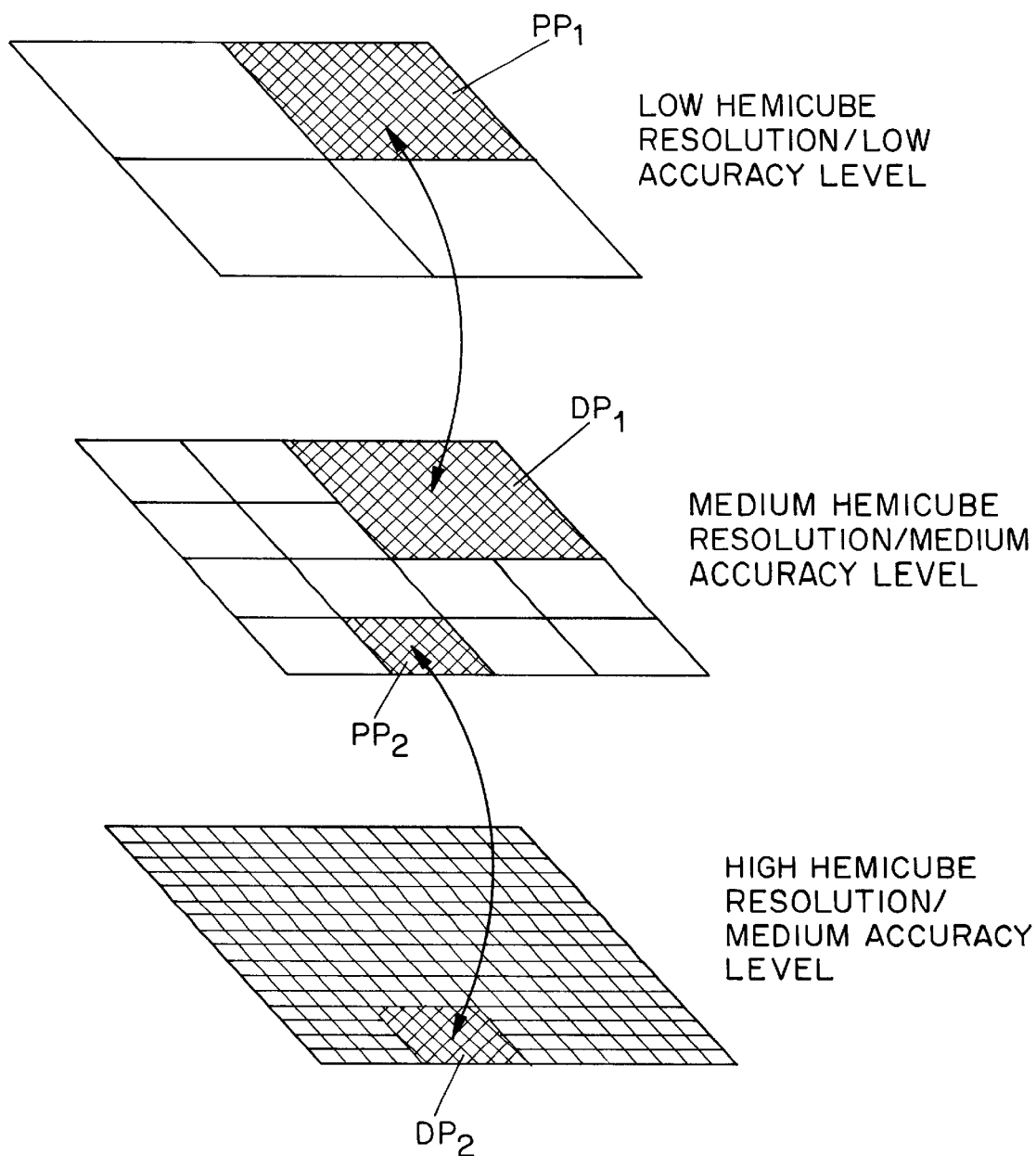


FIG. 15

US 6,172,679 B1

1

VISIBILITY CALCULATIONS FOR 3D COMPUTER GRAPHICS

This is a continuation of U.S. application Ser. No. 08/182,096, filed Jun. 1, 1994, which is incorporated herein by reference.

FIELD OF INVENTION

The present invention relates to computer graphics and, in particular, to the efficient determination of visible and/or invisible surfaces to thereby preferably permit improved hidden surface removal, generally in 3D systems.

BACKGROUND ART

Visible surface detection is one of the most basic operations in 3D graphics. It is applied to generate images of surfaces directly visible to a viewer. Recently, it has also been adopted in the radiosity calculations to compute the energy interactions between surfaces.

The standard strategy of visible surface detection is to divide surfaces into patch elements and compare the spatial relationship between these elements. Using this strategy, the visibility of surfaces cannot be determined until they have been analysed in detail. Although many techniques have been developed to address this issue, none are ideal as they either still require elaborate analysis of surfaces, or they impose various restrictions to the scene.

The limitations of the current techniques can seriously affect the speed of visible surface computation. If the scene is complicated, many surfaces may be invisible. However, the image generation is often slowed down by the need to analyse each surface element in detail.

The same limitation has also seriously affected the efficiency of the radiosity computations. Currently, these computations are very slow due to the need to elaborately compute the visibility between every pair of surface elements. Again, these computations may be substantially reduced if surface elements obviously visible or invisible to each other can be more easily computed.

The early visible surface techniques mainly applied various sorting schemes to find the occluding surface primitives. However, with the advancement in hardware technology, it is now common practice to reduce the need for sorting and comparisons by using a large amount of fast memory. This memory may be used to store the object data, such as a BSP-tree. It may also be used to store the depth and surface projection data, as with a z-buffer.

The z-buffer method is simple and has a very low growth rate. However, it still requires depth evaluations and comparisons at each pixel by all the patches projecting onto it, because their visibility is unknown.

The BSP-tree method has the advantage that if the scene is static, an orderly traversal of the BSP-tree is generally sufficient to establish the depth order. However, it still requires the scan-conversion of each path. In addition, the technique needs to re-compute the BSP-tree whenever objects in the scene move.

There have been two main strategies of avoiding detailed depth analysis of totally invisible entities. One strategy, applies the property that visibility changes can only occur at the contour edges of surfaces. Visibility computations of internal edges or patches can be reduced by first comparing them with these edges.

An alternative strategy is to use the invisible coherence of the scene. These techniques apply the property that an edge

2

is likely to remain invisible in a scan line if it is invisible in the last scan line. Such an edge may therefore be treated specially to avoid unnecessary depth comparisons.

Although the above strategies can reduce some visibility computations, they have limitations. The contour-oriented techniques can operate only in environments which consist exclusively of convex or non-penetrating surfaces. Both the contour-oriented and the scan line approaches are also limited in their ability to reduce visibility computations. In the former, an edge still needs to be tested against the contour edges even if it is invisible. In the latter, all the invisible edge segments at each scan line have to be sorted. These segments also require depth comparisons with the pixels they are on. Finally, all these techniques require some sorting or searching.

SUMMARY OF THE INVENTION

It is an object of the present invention to substantially overcome, or ameliorate, the problems associated with the prior art, through provision of an improved method for performing visibility calculations.

In accordance with one aspect of the present invention there is disclosed a method of reducing the complexity of visibility calculations required for the production of multi-dimensional computer generated images or the reduction of multi-dimensional data to multi-dimensional data having at least one less dimension, said method comprising the steps of:

(1) prior to an occlusion or invisibility relationship computation (known per se) being carried out on a plurality of surfaces, selected viewpoints to be calculated are divided into groups;

(2) for selected ones of said surfaces, determining for each said group whether each said selected surface is

(a) an always unoccluded surface, an always hidden surface, or a remaining surface; or

(b) an always unoccluded surface, or a remaining surface; or

(c) an always hidden surface, or a remaining surface; wherein said remaining surface is a surface which is unable to be determined with certainty as to whether it is either unoccluded or hidden;

(3) exempting from said occlusion or invisibility relationship computation those surfaces which are either always unoccluded or always hidden;

(4) maintaining a record of said remaining surfaces; and

(5) carrying out occlusion or invisibility relationship computations on said remaining surfaces.

In accordance with another aspect of the present invention there is disclosed a method of reducing the visibility related computations in an environment consisting of three-dimensional abstract or physical surfaces, said method comprising the steps of:

(1) prior to a visibility computation (known per se) being carried out, some or all viewpoints or possible occurrences of viewpoints are classified into groups of viewpoints;

(2) defining one or more projection surfaces (known per se) for the purpose of generating simultaneous projections of surfaces or surface elements with respect to a group of viewpoints;

(3) for selected surfaces, and for each group of viewpoints, determining whether those surfaces or their surface elements are always invisible to said group by computing and comparing their simultaneous projections on said projection surfaces;

(4) ignoring or treating specially some or all surfaces or surface elements which are always invisible to said group of

US 6,172,679 B1

3

viewpoints during the actual visibility or visibility related computations for some or all viewpoints in said group.

In accordance with another aspect of the present invention there is disclosed a method of reducing the visibility, radiosity or visibility related computations in an environment consisting of three-dimensional abstract or physical surfaces, said method comprising the steps of:

(1) prior to a visibility computation (known per se) being carried out, some or all viewpoints or possible occurrences of viewpoints are classified into groups of viewpoints;

(2) defining one or more projection surfaces (known per se) for the purpose of the simultaneous projections of surfaces or surface elements with respect to a group of viewpoints;

(3) dividing each of said projection surfaces into regular or irregular grids;

(4) defining a data structure which organizes computer storage for storing of projections on said grids;

(5) for each of the selected surfaces and for each group of viewpoints, simultaneously projecting said surfaces or their surface elements onto the projection surfaces, computing grid cells which are always under the projections, storing the furthest possible distances between viewpoints of said group and said surfaces to provide surface distances corresponding to those cells in said computer storage;

(6) for each patch element of said surfaces, determining those said grid cells that said patch element might project onto, comparing the depths stored in the cells with the furthest possible depth between the patch element and said group of viewpoints to determine whether the patch element is always occluded by other surfaces;

(7) ignoring or treating specially some or all surfaces or surface elements which are always visible to said group of viewpoints during the actual visibility or visibility related computations for some or all viewpoints in said group.

BRIEF DESCRIPTION OF THE DRAWINGS

A preferred and a number of other embodiments of the present invention will now be described with reference to the drawings in which:

FIGS. 1A and 1B show two viewpoint hierarchies;

FIG. 2 illustrates the projections at an arbitrary viewpoint;

FIG. 3 shows the relationship between projection boxes and a fuzzy projection box;

FIG. 4 shows the creation of fuzzy and non-fuzzy regions;

FIGS. 5A, 5B and 5C show a front-facing surface and its corresponding fuzzy and non-fuzzy regions;

FIG. 5D shows a point that is not totally hidden to a viewpoint bounding box;

FIG. 5E shows a point that is totally hidden to a viewpoint bounding box;

FIGS. 6A to 6D show the detection of totally invisible patches;

FIGS. 7A and 7B show the detection of totally visible/non-hiding patches;

FIGS. 8A to 8D show the scan-conversion of the non-fuzzy regions;

FIGS. 8A to 9D show the scan-conversion of the fuzzy regions; and

FIG. 10 illustrates the mapping between a cache and the fuzzy array.

FIGS. 11A and 11B show views of a front-facing surface region;

FIGS. 12A, 12B and 12C show cut-plane views of the projection of surface regions;

4

FIGS. 13A, 13B and 13C show respectively one, two and four room models used in testing the preferred embodiment;

FIG. 14 illustrates the direct computation of the fuzzy region of an edge;

FIG. 15 shows a hemi-cube pyramid structure;

Appendix 1 describes The Detection of Patches Front-Facing to a Viewpoint Bounding Box;

Appendix 2 describes The Computation of the Fuzzy Extent of Boundary Edges and Patches;

Appendix 3 describes The Computation of the Non-Fuzzy Regions;

Appendix 4 describes The Scan-Conversion of the Fuzzy Regions;

Appendix 5 describes Using a Cache to Speed up the Access and Writing of Fuzzy Elements Within the Fuzzy Extent of a Patch;

Appendix 6 describes the direct computation of the fuzzy region of an edge;

Appendix 7 describes scan-conversion of patches on the hemi-cube pyramid; and

Appendix 8 is a list of references.

BEST AND OTHER MODES FOR CARRYING OUT THE INVENTION

The preferred embodiments relate to various methods for calculating three dimensional computer graphic images and they are generally termed herein as the "fuzzy projection methods".

A Simple Fuzzy Projection Method

In this embodiment a method is provided to map the projections of entities at one or more viewpoints to a set of planes. This embodiment also provides a means to compute regions which contain all these projections and regions which are within each and every projection. The spatial position, time and the optical properties described in this embodiment can be replaced by other physical or abstract variables if the replacement does not affect the mathematical relationship of entities in the graphic image.

Prior to detailed discussion of this and the other embodiments, it is useful to introduce various terms and define variables used in 3D graphics.

Firstly, with reference to FIGS. 1A, 1B and 2, a viewer, as shown by the eyeballs in the Figs., is an abstract and dimensionless observer where the visibility of its surrounding objects is to be obtained. A point is visible to a viewer if a ray fired from the viewer to it is not blocked by any opaque object. There can be more than one viewer at any time and each viewer may move or rotate over the time. A viewpoint VP is the spatial position and orientation of a viewer at an instance of time. A view vector VV is a vector from the viewpoint VP to a point PO being observed (see FIG. 2).

Several viewpoints VP can be grouped together through certain properties they share. Related groups of viewpoints may in turn form a higher level group. This merging may be repeated to create a viewpoint hierarchy. A current group is the group of viewpoints VP that are currently being considered. In FIG. 1A, viewpoints VP_1 , VP_2 , VP_3 and VP_4 each reside on a path PA_1 and represent viewpoints at times 1, 2, 3 and 4, and can be considered as one group. Similarly, for the viewpoints VP_5 – VP_7 . Each group of viewpoints is associated with a coordinate system called the group coordinate system CS. A viewpoint bounding box BB of a group of viewpoints VP is the smallest right quadrangular prism enclosing these viewpoints and whose edges are parallel to

US 6,172,679 B1

5

the axes of the group coordinate system. If the positions or occurrences of a group of viewpoints are not precisely defined, the associated viewpoint bounding box should be the said prism which encloses the space likely to be occupied by these viewpoints. A group of viewpoints may be lying on a plane or a point. In such cases, the viewpoint bounding box may degenerate into a plane or a point. A point PO is totally visible from the bounding box BB if it is always visible from every possible viewpoint VP in the box BB. Conversely, a point is totally invisible from the bounding box if it is hidden from the view of every possible viewpoint VP in the box BB. In FIG. 1A, viewpoints VP_1 – VP_4 reside in a bounding box BB_1 , and viewpoints VP_5 – VP_7 reside in a bounding box BB_2 . The boxes BB_1 and BB_2 can be considered as first level bounding boxes. If the boxes BB_1 and BB_2 are combined, a second level bounding box BB_3 is formed. Rather than the bounding box being a square or rectangular prism, it can also be an ellipsoid, sphere or arbitrary volume.

A viewpoint group may be degenerated and contain only a single viewpoint. In this manner, all the techniques applicable to a normal viewpoint group are also applicable to a single viewpoint. In this case, the viewpoint bounding box BB degenerates into a point which coincides with the viewpoint VP.

In FIG. 1B, multiple viewers at different times are illustrated. A path PA_2 of one viewer of a group is shown extending between locations at two points in time. VP_8 represents the viewpoints at time “1” and VP_9 the viewpoints at time “2”, with BB_4 the bounding box at time “1” and BB_5 the bounding box of the union of the viewpoints VP_8 and VP_9 .

Referring now to FIG. 2, shown is a projection box PB, a right quadrangular prism with its centre on a viewpoint VP and its edges parallel to the principal axes of the current group coordinate system CS. The six facets of the box PB are called the projection faces PF. Each projection face PF is on a plane called a projection plane PP. The projection PR of a point PO_1 in the direction perpendicular to a projection plane PP is the intersection between that plane PP and the view vector VV from the viewpoint VP to the point PO_1 . Conversely, since a straight line emanating from the viewpoint VP can only intersect the projection point PR on a projection plane PP, a location on that plane also represents a unique viewing direction from the viewpoint.

In this manner:

$$VV=PO_1-VP$$

The area around the visible space point PO_1 is a visible patch PT and the view vector VV passes through this to an invisible space point IP, occluded by the patch PT.

The projection calculations on each plane are identical to the finding of the image region of an object in the direction perpendicular to that plane. The projection of an object to a projection plane PP therefore represents its image region in the direction parallel to the normal N of that plane.

A viewpoint VP may not be omni-directional but has a limited field of view. Hence certain projection faces may be partially or totally unobservable from the viewpoint. The hidden area of a projection face PF can be obtained by intersecting it with the viewing horizon of the viewpoint VP.

Locations on the projection plane PP can be expressed by a coordinate system CS called the projection coordinates. The origin of this coordinate system is at the centre of the projection face PF. Its x, y axes are parallel to the edges and its z axis is parallel to the normal of the face.

Because they are all parallel, the projection coordinate systems of the current group of viewpoints VP have the same

6

scale factor so that points having the same viewing direction from their viewpoints always have the same projection coordinates.

If points on two surfaces project on the same location, PR for example, on a projection face PF, an occlusion is taking place. The point on the surface further to the viewpoint VP is hidden by the point on the surface closer to the viewpoint VP unless the latter is transparent. This can be determined by the distances d between these points PR and the viewpoint VP.

Every edge of the projection boxes PB of a group is parallel to one of the principal axes (x, y or z) of the respective group coordinate system CS. Since the vectors parallel to the principle axes can have at most six directions, these faces are classified by their normal directions into six sets.

This is shown in FIG. 3 where viewpoints VP_1 , VP_2 and VP_3 , each with respective projection boxes PB_1 , PB_2 and PB_3 , reside within a bounding box BB_1 .

The projections on projection boxes PB_1 – PB_3 can be correlated by another box whose edges are also parallel to the axes of the current group coordinate system CS. This box is called the fuzzy projection box FB. It is called “fuzzy” because it is the combined projection of a single surface (or object, entity, etc.) from all viewpoints, with the combined projection creating a fuzzy image of the surface. Each face of the box is called a fuzzy projection face FF. The face FF lies on a plane called the fuzzy projection plane FP. A projection coordinate system can be defined for a fuzzy projection plane FP and the fuzzy projection face PF on it. Similar to the projection coordinate systems CS of the projection planes, the origin of this coordinate system is at the centre of the fuzzy projection face FF and its axes are respectively parallel to the edges and the normal of the face FF.

By setting the scale factor of the projection coordinate system of each of the fuzzy projection planes to be the same as the scale factor of their associated projection planes PP, each set of projection planes/faces have a one-to-one mapping with the fuzzy projection plane/face that is facing the same direction. Points on a projection plane/face can therefore be linearly mapped to the associated fuzzy projection plane/face.

Points on a set of projection planes having the same projection coordinates, and hence representing the same viewing direction, map to a point on the associated fuzzy projection plane that has the same coordinates. Therefore, similar to the projection planes PP, a point on the fuzzy projection plane FP also represents a unique viewing direction. However, the point is not associated with a particular viewpoint.

Since the viewpoints VP can have a limited field of view, some areas of the projection planes may be unobservable. The hidden area of a fuzzy projection face FF is the area where the corresponding areas on all the associated projection faces are hidden. A fuzzy projection face FF is inactive if all its area is hidden.

To sample the projections on the fuzzy projection box FB, each fuzzy projection plane FP is tessellated by two sets of parallel and evenly spaced grid lines forming a fuzzy array FA. By ensuring that there are always grid lines on the edges of the fuzzy projection faces FF, these faces are divided into identical rectangular or square cells. Each cell is called a fuzzy element FE. Each element FE, in addition to representing the viewing direction associated with its centre, also represents a unique solid angle A of viewing directions, as seen in FIG. 2, represented by its perimeter.

US 6,172,679 B1

7

Although other representation schemes such as the quadtree subdivision can be applied. All the z-buffer oriented operations would be accordingly changed to operation under such schemes, however the regular subdivision of the faces is considered the most efficient embodiment of the present invention.

Surfaces in the viewing environment are approximated by meshes of patches PT. A surface can also be represented by a hierarchy of patches. Different levels of the hierarchy are meshes of patches at different details. Each patch PT is treated as flat and is approximated by a polygon. The projection of a surface SU to a viewpoint VP is the region combining all the projection regions of its patches. This region is called the projection region PE of the surface. Since the projection region PE of a patch PT represents its image region, the projection region PE of a surface SU also represents the image region of that surface.

The smallest right quadrangular prism just enclosing the patch and whose edges are parallel to the axes of the current group coordinate system is called the bounding volume of the patch. The shortest and longest depths of a patch are the shortest and longest depths between the current viewpoints and points on the patch. The depth corresponds to the z-magnitude in the coordinate direction perpendicular to the projection planes currently being considered. As they are usually difficult to find, they are approximated by the shortest and the longest depths between the viewpoint bounding box and the bounding volume of the patch, which can be determined by partitioning the space by the planes of the viewpoint bounding box and determining which partitions the bounding volume of the patch is in.

This is shown in FIG. 4 where a patch PT of a surface is viewed from points VP_1 , VP_2 and VP_3 in a bounding box BB_1 . Associated with each viewpoint VP_1 – VP_3 is a projection region PE_1 – PE_3 which is seen to intersect with the corresponding projection face PF_1 – PF_3 in a different manner for each viewpoint VP_1 – VP_3 .

The projections of the patch PT on the same set of projection faces PF_1 – PF_3 can be mapped to the associated fuzzy projection face FF. The areas on the fuzzy projection box FB_1 containing all these mappings are called the fuzzy (combined) regions FR of the patch because logically the region is fuzzy. They cannot tell whether the patch PT_1 actually projects on the corresponding area on each projection box PB_1 – PB_3 .

Similar to the case of patches, the fuzzy region FR of a surface SU is produced by combining the mappings of the surface projections for a number of patches. In this manner each projection region of the surface is the logical-OR of all the projections of the patches, and the fuzzy projection of each patch is the logical-OR of its projection regions. Accordingly, the fuzzy region FR of a surface SU is also the logical-OR of the fuzzy projections of its patches. This region is equivalent to the superimposition of all the images of the surface seen from the current viewpoints.

On the fuzzy projection box RB_1 , there can also be areas which are always mapped on by the projection regions of the surfaces from the current group of viewpoints. On these areas the logical-AND of the projection regions PE_1 – PE_3 of the surface SU is true. Because the outcomes of the logical-AND operations are always a subset of the outcomes of the logical-OR operations, these areas are always within the fuzzy region FR of the surface SU.

Logically the areas are non-fuzzy as the projection status of the corresponding areas on the current projection boxes is always true. Therefore they are called the non-fuzzy region NF. An alternative term is the umbra region.

8

The non-fuzzy region NF of a surface SU can be obtained by mapping the projections of the surface at all the current group of viewpoints on the fuzzy projection box FB and finding the area within every projection. However, this is costly if there are many viewpoints. To reduce the computations, a series of approximations which err on the side of caution are applied.

It will be apparent to those skilled in the art that there can exist plural non-fuzzy regions for each surface.

Referring to FIG. 5A, the first approximation assumes that the viewpoints can be at any position in the current viewpoint bounding box BB_2 and each viewpoint has a field of view containing all the possible fields of view of the current group of viewpoints. The patches PT_n facing all these viewpoints are determined. They are called the front-facing patches. A method for finding these patches is described in detail in Appendix 1.

Interconnected front-facing patches PT_n are grouped into surface regions called the front-facing sub-surfaces SU_2 . The edges at the borders of these regions are called the boundary edges BE.

Since the front-facing patches PT_n , and their boundary edges BE, are facing all the possible viewpoints VP in the viewpoint bounding box BB, a front-facing sub-surface SU_2 never curves back and is always facing these viewpoints. The projections of its boundary edges BE therefore always surround its projection region PR.

Similar to the patches as shown in FIG. 5B, the projection of each boundary edge BE can be mapped on the fuzzy projection box FB_1 . A fuzzy region FR_1 of the edge BE_1 is the region on the box FB_1 which contains all the projection mappings of that edge at every possible viewpoint in the current viewpoint bounding box BB_2 .

The fuzzy regions of all boundary edges belonging to a front-facing sub-surface can be combined into an area called the fuzzy boundary region BF. If a point is not in the fuzzy boundary region BF, the boundary edges BE do not project on all the associated points on the projection boxes. Because any change of projection status must be at the projection of the boundary edges, the point must be within every mapping of the projection regions of the front-facing sub-surface SU , or it must be outside all the mappings.

Since the fuzzy region is the combined projection of the front-facing sub-surface, any area within it must be mapped by at least one projection region of the sub-surface. Therefore, the areas inside it but outside the fuzzy boundary region BF always contain the mappings of the projection regions of the sub-surface. These areas by definition are the non-fuzzy region NF_1 of the front-facing sub-surface.

As the viewpoints can have any orientation and be anywhere in the viewpoint bounding box BB, the fuzzy region FR of an edge is difficult to compute. However, the extent of this region on a fuzzy projection face can be more easily obtained. The computation of this extent is described in Appendix 2. The smallest rectangle totally enclosing this extent and with boundary on the grid lines of the fuzzy projection plane is called the fuzzy extent EF of the edge on that plane.

This is seen in FIG. 5C where the same fuzzy projection face of FIG. 5B is shown in which the fuzzy boundary region BF_1 is replaced by the fuzzy extents EF, of the boundary edges BE_1 . . . etc.

The region containing all the fuzzy extents EF of the boundary edges always encloses the fuzzy boundary region of the front-facing sub-surface. Therefore, the subtraction of it from the fuzzy region FR produces an area always within the non-fuzzy region. This area is used to approximate the non-fuzzy region NF.

US 6,172,679 B1

9

If a surface contains large patches and hence large boundary edges, the use of the fuzzy extents to approximate its fuzzy boundary region may be inefficient. The fuzzy regions of its boundary edges can be directly evaluated using the method described in appendix 6. Alternatively, each edge can be divided into sub-edges. The fuzzy region of that edge can then be replaced by the fuzzy extents of its sub-edges. An Invisibility Fuzzy Projection Method for the Detection of Totally Invisible Patches.

In this embodiment a method is provided to compare the mappings of entities on the fuzzy projection faces FF associated with a group of viewpoints VP. Through this operation, entities which may be treated as invisible to all these viewpoints, can be detected.

Referring to FIG. 6A, a viewpoint boundary box BB_3 is shown to observe three patches PA_A , PT_B and PT_C through a mesh surface SU_M . FIGS. 6B, 6C and 6D show respectively the fuzzy extents EF_A , EF_B and EF_C for the patches PA_A , PT_B and PT_C on respective fuzzy projection faces FF_{A-C} . Also shown is the non-fuzzy region NF_3 of the mesh surface SU_M .

The determination of totally invisible surfaces is based on a property of the non-fuzzy regions. In FIG. 5D, point P02 is hidden by surface SU1 from viewpoint VP11 in the current viewpoint bounding box BB_5 . If the point is visible from another viewpoint VP10 which is also in BB_5 , then SU1 has to curve back and reach behind P02. However, in such a situation part of SU1 next to a point SP would become back-facing to viewpoint VP12 in BB_5 . SU1 would then contain several front-facing surfaces which do not contain the surface around P02. The non-fuzzy region of SU1 then would not cover the projection mapping of P02.

Because of the above property, if the projection mapping of a point from an arbitrary point in the current viewpoint bounding box is within the non-fuzzy region of a surface, and if that point is behind the surface from the viewpoint, that point would always be hidden by the surface from all viewpoints in the box. This is shown in FIG. 5E. Here, the invisibility detection is performed using a single viewpoint (either one of VP13 or VP14 or other in the box BB_6).

The above observation can be used to detect totally invisible patches, each patch whose visibility is to be determined is selected. The fuzzy elements within the projection mapping of the patch from a viewpoint in the viewpoint bounding box are accessed. If the distance stored in each element is smaller than the depth of the patch from the viewpoint in the direction associated with that element, the patch is an always invisible patch because of the property of non-fuzzy regions just mentioned.

To reduce the above computations, the projection mapping of the patch can be substituted by the extent of that mapping, or the fuzzy extent of the patch. Alternatively, the depth of the patch from the viewpoint may be replaced by the longest possible depth between the patch bounding box and the viewpoint or the viewpoint bounding box. Such approximations always err on the side of caution and do not affect the correctness of hidden surface computations.

A 2D fuzzy arrays FA_A , FA_B , FA_C , with as many storage elements as the fuzzy elements EF on each active fuzzy projection face FF_{A-C} are allocated to store the mapping information on the face. All the fuzzy arrays FA_{A-C} are collectively called a fuzzy buffer generally, a memory that is two-dimensionally addressable.

Each fuzzy element EF contains a depth field. The depth field stores the depth of the patch PT whose fuzzy extent EF surrounds the fuzzy element FE.

A cycle of fuzzy buffer computations is applied for each group of viewpoints. First, the depth fields are initialized to

10

infinity to indicate that no surface has yet been projected on the projection box PB.

The fuzzy extents EF of the boundary edges BE and the fuzzy extents EF of the patches PT of a front-facing sub-surface of selected opaque surfaces are computed (as described in Appendix 2). Usually, large surfaces are selected. Based on these extents, the non-fuzzy region NF of the front-facing sub-surface is computed by a scan line technique described in Appendix 3. Furthermore, the fuzzy region of an edge can be found by direct computation as described in Appendix 6.

Using the above, the longest depth of the fuzzy elements in each horizontal segment of every non-fuzzy region are obtained. The distance stored in each element of the array is compared with the depth stored in the corresponding element in the fuzzy buffer. If the latter is larger, it is replaced by the former.

After all the non-fuzzy regions of the opaque front-facing sub-surfaces have been updated to the fuzzy elements, each element in the fuzzy buffer contains the longest depth of the closest surface that can be seen from a chosen viewpoint in the current viewpoint bounding box in the direction associated with that element. The fuzzy buffer can then be used to compute patches totally invisible from the current viewpoint bounding box. These patches are deemed to be totally invisible patches.

To detect totally invisible patches, the fuzzy elements within the fuzzy extent, the projection mapping, or the extent of the mapping, of each patch are accessed. If the depth stored in each element is smaller than the shortest depth of the patch, the patch is always hidden in the direction represented by that element. Because of the property of non-fuzzy regions, or because of the property that the fuzzy extent covers all the possible viewing directions of the patch, the patch is totally invisible if it is found to be hidden at every fuzzy element within that extent.

As seen in FIG. 6B, the fuzzy extent EF_A of patch PT_A falls entirely within the non-fuzzy region NF_3 . Accordingly, the patch PT_A is hidden from the viewpoints in the bounding box BB_3 .

In FIG. 6C, the fuzzy extent EF_B of patch PT_{VB} both inside and outside the non-fuzzy region NF_3 . Accordingly, those four elements outside may be either visible or hidden, and for the six elements inside, for all view directions through those elements, the patch PT_B is hidden.

In FIG. 6D, the fuzzy extent EF_C of patch PT_C passes the depth test for every fuzzy buffer element it falls on. Patch PT_C is always invisible from the bounding box BB_3 .

Since there is always substantial overlappings between the fuzzy extents of adjacent patches, a high-speed cache can be used to speed up the reading and writing of the fuzzy elements within the fuzzy extent of each patch. The cache can also be used in the visibility fuzzy projection technique discussed in the next section. Appendix 5 describes in more detail the use of a cache memory.

A Visibility Fuzzy Projection Method for the Detection of Totally visible Patches.

In this embodiment a method is provided to compare the mappings of entities on the fuzzy projection faces associated with a group of viewpoints. Through this operation entities which may be treated as visible and not hiding other entities to all these viewpoints can be detected.

For simplicity and speed, this method assumes that every surface does not hide itself in any viewing direction from the current viewpoint bounding box except those parts which do not face that direction. This implies that no two parts of the same surface that are facing a viewpoint in the bounding box

US 6,172,679 B1

11

would hide each other. All the planar and quadric surfaces have this property. Other surfaces can be subdivided into smaller surfaces which satisfy this criterion.

As seen in FIG. 7A, a viewpoint bounding box BB_4 observes three patches PT_D , PT_E and PT_F arranged about a mesh surface SU_N .

Similar to the technique for detecting totally invisible patches, a fuzzy buffer consisting of several arrays is used to store the results of projections on the fuzzy elements.

Every element of the fuzzy buffer contains a field called the homogeneous indicator. This is a three-valued field which indicates whether there is any projection on the associated grid cell on the fuzzy projection box and whether one or more surfaces are being projected onto the cell. The indicator is zero if it is not enclosed by any fuzzy region. The indicator is one if it is inside the fuzzy region of one surface. The indicator is two if it is within the fuzzy regions of more than one surface.

First, the homogeneous indicators are initialized to zero. This shows that no surface has yet been projected on the fuzzy buffer.

Using the scan-conversion technique described in Appendix 4, the horizontal segments of the fuzzy region of each surface are obtained.

The homogeneous indicators under each segment are examined. Each indicator is incremented by one if it is zero or one.

After the fuzzy regions of all the surfaces have been updated into the fuzzy buffer, each patch is examined. It is ignored if it has been detected as totally invisible by the invisibility fuzzy projection technique.

If the homogeneous indicators within the fuzzy extent of the patch are all one, only one surface can be seen from the current viewpoint bounding box in all viewing directions within that fuzzy extent. The surface has to be the one the patch is on. The patch therefore does not hide other surfaces. Since the front-facing parts of the surface do not hide each other, the patch is always visible unless when it is back-facing. The patch is called a totally visible/non-hiding patch.

If some of the homogeneous indicators within the fuzzy extent of the patch are not one, the patch might be covering or be occluded by patches of other surfaces at certain viewpoints in the current viewpoint bounding box.

This is shown in FIG. 7B where the fuzzy projection face FF_4 is shown with each element having either the value 0, 1 or 2. As indicated for the patches PT_D and PT_F and their corresponding fuzzy elements FE_D and FE_F , these are not totally visible and not hiding other patches as some of the fuzzy elements within their respective fuzzy extents are mapped to more than one surface. Patch PT_E is totally visible/non-hidden as all its fuzzy elements FE_E within its extent are mapped only to one surface (i.e. SU_N).

MODIFIED VISIBILITY FUZZY PROJECTION METHOD

The above technique scan-converts the fuzzy regions of each surface twice because each fuzzy element may be overlapped by the fuzzy regions of several patches of the same surface. This can be avoided by using a linked list called the projected patch list to store the overlapped patches for each fuzzy element. Other data structures such as an array can also be used in place of the linked list. The modified technique consists of the following steps:

1. As the original technique, each fuzzy element contains a homogeneous indicator and a surface ID field. It also contains a projected patch list. The homogeneous indicator and the surface ID fields are initialized to zero and null

12

respectively to indicator that no surface has been projected on the fuzzy element. The projected patch list associated with the element is initialized to the null list.

2. Each patch contains a totally visible indicator. This indicator is initialized to ON. Each patch in the environment is accessed. The fuzzy region of each patch is computed and each element within it is accessed.

3. If the homogeneous indicator of each fuzzy element accessed is zero, the fuzzy element has not been projected onto by the fuzzy regions of any surface. The surface ID of the current patch is written to the surface ID field. The homogeneous indicator is set to one. If the totally visible indicator of the current patch is ON, its ID is added to the projected patch list of the current fuzzy element. Otherwise there is no such necessity because the projected patch list is used to reset the totally visible indicators of patches.

4. If the homogeneous indicator is one, the surface ID of the current patch is compared with the surface ID stored in the surface ID field of the current fuzzy element. If they are the same and the totally visible indicator of the patch is ON, the ID of that patch is added to the projected patch list associated with the current fuzzy element.

If the ID's of the two surfaces are different, the homogeneous indicator is set to two to indicate that there is more than one surface projecting onto the current fuzzy element. The current patch and all the patches with ID's stored in the projected patch list of the current element may not be totally visible. Therefore, their totally visible indicators are set to OFF. Records in the projected patch list can be removed after all their totally visible indicators have been updated.

5. If the homogeneous indicator is two, the current fuzzy element has already been found to be projected onto by more than one surface. The current patch therefore may not be totally visible. The totally visible indicator of its record is set to OFF.

6. After the processing of all patches, a patch is determined to be totally visible if its totally visible indicator has not been set to OFF.

The pseudo-code of the above processing is shown below:

```
initialize the totally invisible indicators of all patches to ON;
initialize the homogeneous indicator of all fuzzy elements to
zero;
clear all projected patch lists of the fuzzy elements,
for (each patch of the current surface)
do
  for (each fuzzy element under the fuzzy region of the patch)
  do
    if (the homogeneous indicator is zero)
    then
      write the surface ID of the current patch to the surface ID
      field;
      set the homogeneous indicator to one;
      if (the totally invisible indicator of the current patch is
      not OFF)
      add the patch ID to the projected patch list of the
      fuzzy element;
    endif
    else if (the homogeneous indicator is one)
    then
      if (the surface ID of the current patch
      is the same
      as the surface ID stored in the fuzzy
      element)
      then
        if (the totally invisible indicator of
        the current
        patch is not OFF)
```

US 6,172,679 B1

13

```
-continued
    add the patch ID to the projected
    patch list of
    the fuzzy element;
    endif
else
    set homogeneous indicator to two;
    set the totally invisible indicator
    of the
    current patch and each patch
    indexed by the projected patch
    list of the fuzzy element to
    OFF;
    endif
else
    set the totally invisible indicator of
    the current patch to OFF;
    endif
done
done
```

Use of the Visibility and Invisibility Methods for the Radi-
osity Methods

By the application of the foregoing methods, a list of the totally invisible patches and the totally visible/non-hiding patches can be obtained for a viewpoint group. The visibility of these patches remains the same for all the subgroups and viewpoints under that group. Therefore these patches need not go through elaborate visibility computations in these subgroups and viewpoints.

For each viewpoint in the group, the totally invisible patches may be ignored. The totally visible/non-hiding patches need not be compared with other patches to determine their visibility and form-factors. Their form-factors can be computed directly from their orientations, positions and shapes using mathematical formulas describing the radiative transfer between surfaces.

If a patch is bright or close to the current group of viewpoints, the direct form-factor computations need to be carried out in higher accuracy. The patch can be subdivided into smaller patches and the form-factor computations are carried out for these sub-patches instead. Alternatively, accurate radiative transfer formulas such as the Nussel analog technique can be used. This technique finds the fractional area of the patch after it has been projected onto the surface and then the base of a hemisphere whose centre is at the viewpoint.

If accurate form-factor computations are not required, the form-factor of a patch for the current group of viewpoints may be obtained by assuming that it is constant throughout that patch. The standard form-factor equation can be simplified to:

$$F_{\cos A} \cos B A / (\pi r^2)$$

where A and B respectively are the angles between the view vectors and the normals of the observed patch, and the patch the viewpoint is on, and r is the distance between the viewpoint and the centre of the observed patch.

The fuzzy projection methods require orderly top-down traversal of viewpoints according to their hierarchy. However, in techniques such as the progressive refinement methods, the receiving patches are usually accessed in decreasing order of their brightness. Therefore, if these techniques are used in conjunction with the fuzzy projection methods, the form-factor computations of some viewpoints in a group may not have been carried out since not all the viewpoints within the group may be processed in one go.

14

Information about which are the totally visible/non-hiding patches and which are the totally invisible patches of the partially completed group has to be stored in memory. The outstanding viewpoints in the group can then use this information when their turn to be processed has arrived. This information can be removed when all the viewpoints in the group have been processed.

In techniques such as the progressive refinement methods (known per se), the form-factors between patches have to be repeatedly computed. Therefore, if memory permits, the information about which patches are totally visible/non-hiding and which patches are not totally invisible may not be removed and can be advantageously used.

In the first round of computations, all the fuzzy projection and normal hemicube computations for every receiving patch are carried out as required. During these computations, the emission patches at each group/viewpoint are classified into three groups: totally visible/non-hiding, totally invisible, and the remainder. This classification information is stored using data structures such as arrays or linked lists.

In subsequent rounds of computations, where the form factors between patches have to be re-evaluated, the fuzzy projection computations are not repeated. The totally visible/non-hiding and the totally invisible patches of each-viewpoint can be simply retrieved from the stored information.

The memory for storing the classification of patches for each group of viewpoints can be reduced by several methods. First, since patches found to be totally visible/non-hiding or totally invisible from a group will maintain the same status for all its subgroups, they need not be repeatedly stored.

Secondly, because the union of the three lists of patches is the set of all the patches that need to be classified for each group of viewpoints, only two lists need to be stored for the group. To further reduce storage, the smallest pair of the three lists can be kept.

Also, because the records of patches are usually orderly organized and accessed, the patch IDs in the lists usually have some order. The differences between the IDs of successive patches in the lists are often much smaller than the magnitude of these IDs. Whenever this occurs, these differences instead of the patch IDs can be stored. A sign bit can be used to distinguish between the ID's and difference values. The list can be sorted by the ID for more efficient use of this arrangement.

Finally, the visibility status of patches often do not change much from one group of viewpoints to an adjacent group of viewpoints. Therefore, using the patch lists of a viewpoint group as a starting point, a series of nearby groups can store their patch lists incrementally. Starting from the first group, each adjacent group only needs to store the lists of patches that need to be deleted from, and added to the patch lists of the previous group to form the current lists. Again, a special field can be used to distinguish the use of this arrangement.

ENHANCING THE ACCURACY OF FORM-
FACTOR COMPUTATIONS.

The fuzzy projection techniques can also be used to achieve optimal accuracy of the form-factor computations. After the filtering off of totally invisible patches by the invisibility technique, the visibility technique detects totally visible and non-hiding patches. As mentioned above, different methods which yield different accuracy in the computations of the form-factors of these patches can be applied.

The remaining patches are not totally invisible. They are also not totally visible/non-hiding. The accuracy of their form-factor computations can be determined by the following steps:

US 6,172,679 B1

15

a. First, several levels are defined such that each of them corresponds to a level of accuracy to be reached by the form-factor computations of patches. Each accuracy level determines the strategies of form-factor computations such as the resolution of the hemicube buffer, the level of details of patches, whether to use ray-tracing instead of the hemicube buffer, or the number of rays traced per sampling point.

b. All the patches not found to be totally invisible or totally visible are classified by their accuracy level according to their brightness, importance, distance from the current group of viewpoints, and other properties. Each patch record has a field which stores its accuracy level. An accuracy level field is also allocated for each fuzzy element in the fuzzy buffer. Each field is initialized to the lowest accuracy level.

c. Patches are accessed in the order such that those having higher accuracy levels are scan-converted first.

For each patch accessed, the same scan-conversion of their fuzzy extents applied in the visibility fuzzy projection computations is carried out. However, instead of the homogeneous indicators, the accuracy level fields in each fuzzy element access is examined. If the value in a field is larger than the accuracy level of the patch, the patch is likely to be hiding or be hidden by a patch at a higher accuracy level. The accuracy level of the patch is set to this value. If the value is smaller than the original accuracy level of the patch, the later is written to the former.

d. After the processing of all patches, the updated accuracy level of each patch shows the maximum level of accuracy its form-factor computations needs to be carried out for the current group of viewpoints. If this level is higher than the accuracy the patch can provide, it may be recursively subdivided and replaced by sub-patches whose accuracy matches the level.

If the actual surface of a patch is curved, the projections of its sub-patches may be outside its projection. Therefore, for the subdivision strategy to be effective, the fuzzy extent of each entity should contain the fuzzy extents of all its sub-entities during all the fuzzy projections. It can be approximated by the fuzzy extent of the axis-aligned bounding box containing all these sub-entities.

e. The accuracy level field of a fuzzy element indicates the accuracy of the computations that needs to be carried out in the corresponding viewing directions from the current group of viewpoints or the corresponding regions on the hemicubes of these viewpoints. The form-factor of a patch can be computed in variable resolutions which match the different accuracy requirements of the fuzzy elements covered by that patch. This can be done in either the ray-tracing or the hemicube approach of form-factor computations.

f. In the ray-tracing approach, the form-factors of patches at a viewpoint is computed by tracing rays in all observable directions. For each of the current group of viewpoints, the number of rays to be fired within a solid angle associated with a fuzzy element depends on the accuracy level of that element. If the level is high, more than one ray should be traced. Conversely, if the accuracy levels of several adjacent fuzzy elements are low, only one ray may be traced for the directions associated with these elements.

g. In the hemicube approach, different regions on the hemicube should have different sampling resolutions which match the accuracy requirements of the fuzzy elements associated with these regions. This can be achieved by the use of a pyramidal representation called the hemicube pyramid. The scan-conversion of patches on the hemicube pyramid is described in Appendix 7.

16

Use of the Visibility and Invisibility Methods for Hidden Surface Computations

In this embodiment a method is provided to apply the computed results of the visibility and invisibility fuzzy projection methods in the hidden surface computations.

In shadow computations, in applications where a series of images are to be taken, such as computer animation, flight simulation, or dynamic graphics, the hidden surface removal operations need to be repeated many times. The position of the viewer when an image of the surrounding is taken may be treated as a viewpoint. The viewpoints may also be combined into a hierarchy of viewpoint groups.

If the combined field of view of the current viewpoint group is narrow, the projection boxes and the fuzzy projection box can be flattened so that only a fuzzy projection plane and its associated projection planes are active for the current group of viewpoints.

By using the invisibility fuzzy projection method, patches found to be totally invisible to a group of viewpoints, do not need to be considered for the subgroups and the individual viewpoints in the group.

By using the visibility fuzzy projection method no depth computations and comparisons of totally visible patches during the scan-conversion is necessary.

Use of the Visibility and Invisibility Methods for the Ray Tracing Computations

In this embodiment a method is provided to apply the computed results of the visibility and invisibility fuzzy projection techniques in the ray tracing techniques.

In a ray tracing application, the objects and surfaces in the environment are often organized as a hierarchical geometric model. If the number of rays traced from a node of the geometric model justifies the use of the fuzzy buffer methods, a viewpoint group is created by treating every point on the surfaces of that node where a ray might emit as a viewer. The position of the viewer at any instance of time is a viewpoint. The viewpoints can be grouped into a hierarchy of viewpoint groups.

From the fuzzy buffer computations, patches which are totally visible/non-hiding to a group of viewpoints corresponding to each node and patches which are totally invisible to this group of viewpoints.

When a ray is traced from a node, all the totally invisible patches need not be considered. Also, the ray is tested first with the totally visible/non-hiding patches. If the ray hits one of those patches, no further testing with other patches need to be carried out as the patch being hit is the closest one on the path of the ray.

If both the radiosity method and the ray tracing techniques are used, the results of the fuzzy buffer computations can be used for both methods.

USE OF THE VISIBILITY AND INVISIBILITY FUZZY PROJECTION TECHNIQUES IN COMPUTER VISION.

The visibility and invisibility fuzzy projection techniques can be used in computer vision. The occlusion analysis of objects is a vital operation in computer vision. Based on this analysis, hypotheses are made to construct a 3D model. This model is then matched with the vision data or existing models. It may be repeated matched and refined until it is acceptable. Usually there are a lot of uncertainty and vagueness in the vision data and the hypothesis. Such imprecision can be accommodated by the present techniques in the following ways:

1. During the model construction phases, a viewpoint bounding box can be defined which include all the likely positions of the viewpoint.

US 6,172,679 B1

17

2. During the model construction phase, a 3D model is generated based on the vision data. However, if the exact locations and shapes of entities such as edges and patches are uncertain, the entities may be approximated by the bounding boxes which contain all their likely locations.

3. Based on the hypothesized model, the invisibility and visibility techniques are carried out to compute totally visible surfaces, totally invisible surfaces and surfaces whose visibility cannot be determined.

4. The totally visible surfaces correspond to the vision data whose information is relatively explicated. Such data may be treated as successfully interpreted or more cursorily checked. Areas on the fuzzy projection box which are projected onto by surfaces whose visibility cannot be determined correspond to the vision data which are more obscured. The areas on the image plane corresponding to these areas are further analysed.

5. In the next round of analysis, the hypothesis becomes more refined. The viewpoint bounding box and bounding boxes of entities may be accordingly shrunked.

The preferred embodiment has been implemented on a general purpose computer adapted via programming in the C-language and tested using a set of models which differ in depth complexity. Three of the models are shown in FIGS. 13A, 13B and 13C. The visible surface computations have been computed for five nearby viewpoints. The times of hidden surface computations under different pixel and patch resolutions have been measured and are shown below.

EXAMPLE 1

Coarse patches, coarse screen pixels (400×400 pixels), resolution of fuzzy buffer: 400×400 pixels

Model	Normal Hidden Surface Removal		Fuzzy Hidden Surface Removal		Patches invisible to all viewpoints
	CPU time for 5 viewpoints (sec)	Total Patches	CPU time for fuzzy computations (sec)	CPU time for 5 viewpoints (sec)	
1 room	11.2	10807	1.2	11.1	157
2 room	21.8	19137	2.1	17.3	3942
3 room	35.7	27467	2.9	23.9	9041
4 room	46.3	35797	3.6	25.6	16140

EXAMPLE 2

Coarse patches, fine screen pixels (1000×1000 pixels), resolution of fuzzy buffer: 400×400 pixels

Model	Normal Hidden Surface Removal		Fuzzy Hidden Surface Removal		Patches invisible to all viewpoints
	CPU time for 5 viewpoints (sec)	Total Patches	CPU time for fuzzy computations (sec)	CPU time for 5 viewpoints (sec)	
1 room	20.7	10807	1.2	20.7	157
2 room	40.8	19137	2.1	32.4	3942
3 room	56.4	27467	2.9	37.8	9041
4 room	72.9	35797	3.6	40.0	16140

18

EXAMPLE 3

Fine patches, coarse screen pixels (400×400 pixels), resolution of fuzzy buffer: 400×400 pixels

Model	Normal Hidden Surface Removal		Fuzzy Hidden Surface Removal		Patches invisible to all viewpoints
	CPU time for 5 viewpoints (sec)	Total Patches	CPU time for fuzzy computations (sec)	CPU time for 5 viewpoints (sec)	
1 room	25.0	21263	2.0	24.1	1605
2 room	41.1	38293	2.5	25.0	15301
3 room	60.6	55323	3.4	29.1	28712
4 room	80.7	72353	4.6	33.8	42103

EXAMPLE 4

Fine patches, fine screen pixels (1000×1000 pixels), resolution of fuzzy buffer: 400×400 pixels

Model	Normal Hidden Surface Removal		Fuzzy Hidden Surface Removal		Patches invisible to all viewpoints
	CPU time for 5 viewpoints (sec)	Total Patches	CPU time for fuzzy computations (sec)	CPU time for 5 viewpoints (sec)	
1 room	42.0	21263	2.0	40.5	1605
2 room	75.9	38293	2.5	45.4	15301
3 room	111.3	55323	3.4	53.8	28712
4 room	148.3	72353	4.6	61.9	42103

The above results indicate that the overhead of the preferred embodiment is low. It also indicates that substantial computation savings can be achieved when the depth complexity of the models is high.

There are a number of advantages to be gained by applying the strategies in the disclosed embodiments. Firstly, they do not have the restrictions encountered by the earlier hidden surface algorithms. For example, they do not preclude the presence of intersecting surfaces or patches in the environment. In addition, surface patches can be curved and the scene need not be static. The method also has very little computational overhead. Because it operates on a z-buffer, it can be easily implemented in hardware.

The preferred embodiment can be further enhanced using heuristics. In applications such as the interactive walk-through of the interior of a building, the surfaces most likely to form large crisp (non-fuzzy, umbra) regions are those representing walls, floors and ceilings. Therefore, only the crisp regions of these surfaces need to be scan-converted to the fuzzy buffer.

In addition to reducing hidden surface removal computations, the preferred method can also be used in the radiosity method. Currently the radiosity method is very slow, mainly because it needs to compute the visibility of surfaces for a large number of surface points to determine the form-factors. However, by grouping these points and treating them as groups of viewpoints, the preferred embodiment can be used to determine surfaces totally visible to a group of these points. The fuzzy computations for each viewpoint bounding box is apportioned among all viewpoints in the box. Because there would be viewpoint for

every surface point, the density of viewpoints in each viewpoint bounding box could be very high. Therefore, the advantage of using the method in the radiosity method can be more than in the normal hidden surface removal.

The methods disclosed can also be used in virtual reality applications and other applications where abstract data is utilized. In the future such an application would need to display complex and dynamic environments in real time using the radiosity method. With the addition of a time dimension, the density of viewpoints in each viewpoint bounding box could be an order of magnitude higher than that of the radiosity method. Consequently, the methods disclosed could substantially speed up these applications; Other applications include the manipulation of physical data such as the generation, processing and display of scientific data such as energy spectra data.

The foregoing describes only a number of embodiments of the present invention and modifications, obvious to those skilled in the art, can be made thereto without departing from the scope of the present invention.

APPENDIX 1

The Detection of Patches Front-Facing to a Viewpoint Bounding Box

Referring to FIG. 2, whether a patch PT is facing a viewpoint VP can be determined by the angle A between the normal N of the patch PT and the view vector VV. The latter is the vector from the viewpoint VP to the centre of the patch PT. The cosine of the angle is:

$$\begin{aligned} \cos A &= N \cdot (PO - VP) / k \\ &= k(Nx(x' - x) + Ny(y' - y) + Nz(z' - z)) \\ &= k(Nx x' + Ny y' + Nz z' - \\ &\quad Nx x - Ny y - Nz z) \end{aligned}$$

where VP=(x, y, z) is the position of the viewpoint, N=(Nx, Ny, Nz) is the unit normal of the patch, and PO=(x', y', z') is the position of the patch centre. k is a positive coefficient which turns the view vector (VV=PO-VP) into a unit vector.

If the patch PT is facing the viewpoint bounding box BB, the angle A must be obtuse. Hence cosine A must be negative for every possible viewpoint VP in the box. To check this, the maximum value of the cosine of A is found. If it is positive, then the patch would be back-facing to certain viewpoints in the bounding box.

Since PO and N are constant vectors, the maximization of cosine A requires each of Nx x, Ny y and Nz z to be minimized. Depending on the signs of the components of N, either the maxima or the minima of x, y and z are used. These coordinates correspond to the locations of the corners of the current viewpoint bounding box.

If a patch moves or rotates over time and the viewpoint bounding box contains viewpoints at different times, then the smallest cosine A at each instance of time may be computed and the smallest of them chosen.

Patches with bounding volumes inside or intersecting the viewpoint bounding box are always treated as not front-facing. These patches can be detected by an initial bounding volume check.

FIG. 11A shows a cut-plane view of a surface region SU_R front-facing a set of viewpoints VP₇, VP₈, VP₉ and FIG. 11B shows the same surface region front-facing to a viewpoint bounding box BB_R. Since the front-facing-patches PT_F are

facing all possible viewpoints in the box BB_R, the region SU_R never curves back and is always facing these viewpoints. The view projections of the border of the region SU_R therefore always enclose its projection. This property can be used in subsequent fuzzy projection computations.

FIGS. 12A-12C show cut-plane views of projections of surface regions front facing to a group of viewpoints and their respective bounding boxes.

In FIG. 12A if all patches PT_A in a surface region are front facing, the projections BP_A of the boundary edges of the region always surround the projection PE_A of that region.

In FIG. 12B if some surface patches PT_B in a surface region are not front facing, the region may curve back. The projections BP_B of the boundary edges of the region may be inside the projection PE_B of that region.

In FIG. 12C the projection BP_C of the boundary of each front-facing region of a convoluting surface still encloses the projection PE_C of the region.

APPENDIX 2

The Computation of the Fuzzy Extent of Boundary Edges and Patches

As shown in FIG. 2, the projection coordinates PR (X,Y) of a point PO (x',y',z') to a viewpoint VP (x,y,z) on the projection plane PP facing the positive z direction is

$$X = (x' - x) * d / (z' - z) \tag{1}$$

$$Y = (y' - y) * d / (z' - z) \tag{2}$$

where d is the shortest distance from the viewpoint to the projection plane facing the z direction.

To maximize X and Y, the minimum values of x and y are used. If X and Y are to be minimize, the maxima of x and y are used. Since z is part of the divisors, whether to use its maximum or minimum value depends on the signs of (x'-x) and (y'-y) after choosing the extreme values of x and y. Since the edges of the viewpoint bounding box are parallel to the axes, viewpoints having these extreme x, y and z values are at the corners of the box.

According to equations (1) and (2), the points which have the extreme values in x', y' and z' would yield the maximum and minimum projections. However, these points might not be on the current projecting entity, which can be a boundary edge or a patch.

To overcome the problem, the bounding volume of the entity is used to approximate that entity. Its projections can be used as an estimation as they enclose the projections of that entity.

Sometimes an entity may be subdivided into sub-entities. Its position and shape may also be non-static over time or uncertain. Hence the bounding volume of an entity should be a volume that encloses all the space it is likely to occupied, and all the bounding volumes of its sub-entities that are likely to be considered.

Hence the bounding volume of an entity should enclose all its sub-entities that are likely to be considered in the computations.

Following the derivations above, the maxima of x' and y' and the minima of x and y are used to maximize the projections. Whether to maximize or minimize (z'-z) depends on the signs of (x'-x) and (y'-y). If these terms are positive, the term should be minimized. Hence the smallest z' and the largest z are used. Otherwise, the largest z' and the smallest z are instead chosen.

The finding of the minimum projections is the opposite of maximization. Hence the decisions to minimize or to maximize the individual terms are accordingly changed.

US 6,172,679 B1

21

Having obtained the projections which give the maximum and minimum X and Y values, these projections are mapped to the fuzzy projection plane FP whose normal is toward the z direction.

A rectangle on the plane with these projections as corners and with edges parallel to the edges of the corresponding fuzzy projection face FF can be defined. They are either rounded up or down to create another region which contains the original rectangle and all the pixels under it. This is the fuzzy extent EF of the patch on the fuzzy projection plane FP.

By rotating the axes, the same equations (1) and (2) can be used for projection planes facing other directions if the terms in equations (1) and (2) are accordingly permuted. The fuzzy extents corresponding to these planes can then be similarly obtained.

If a patch is within the viewpoint bounding box, then for some viewpoints z' is smaller than or equal to z in equations (1) and (2). The fuzzy extents would then extend to infinity in the directions which can be determined by the signs of the denominators and numerators in these equations.

APPENDIX 3

The Computation of the Non-Fuzzy Regions

To compute the non-fuzzy region, it is first necessary to obtain its relationship with the fuzzy boundary region and one of the projection regions.

Consider two viewpoints in the current viewpoint bounding box and a point on a fuzzy projection plane. Assume that the projection of the front-facing sub-surface at one viewpoint maps on that point and at the other viewpoint does not map on that point.

Because the change of projection status can only occur at the projections of boundary edges, on a curve joining the two points there must exist another viewpoint such that the projection of a boundary edge from it would map on the same point on the fuzzy projection plane. Since this curve can be totally inside the current viewpoint bounding box, that point must be within the fuzzy region of the boundary edges for that bounding box.

If a point on a fuzzy projection plane is within the mapping of a projection region of the front-facing sub-surface, and if it is outside the fuzzy region of the boundary edge, then the point is always within the projection regions of the sub-surface for all the viewpoints in the current viewpoint bounding box. The point is by definition inside the non-fuzzy region. Since the fuzzy boundary region is always within the area containing the fuzzy extents of boundary edges, the point is still inside the non-fuzzy region if the former is approximated by the latter.

Based on the above, the approximated non-fuzzy region can be obtained by finding the projection region of the front-facing sub-surface from a viewpoint in the current viewpoint bounding box, projecting it to the fuzzy region, and subtracting all its area intersected with the fuzzy extents of the boundary edges. A modified scan line algorithm may be used for this task.

First, an arbitrary viewpoint in the current viewpoint bounding box is selected. Usually the centre of the bounding box is chosen. This viewpoint is called the chosen viewpoint. The projection coordinates of the edges in the front-facing sub-surface from this viewpoint are found. Since the projection coordinate system and the fuzzy projection coordinate system have one-to-one mapping, the mapping of the former to the latter is trivial.

22

This is followed by the computations of the fuzzy extents of the patches described in Appendix 2.

An illustration of such an arrangement is shown in FIG. 8A where two patches A and B intersect a scan line SL and have respective fuzzy extents EF_A and EF_B of their boundary edges.

Two active edge lists are maintained during the processing of scan lines. The first list contains the active edges of the patches. The second list contains the pairs of active vertical edges of the fuzzy extents of boundary edges.

In FIG. 8A, the first active edge list is:

A1 A2 B1 B2

the second active edge list is:

E1 E2 E3 E4

During the processing of successive scan lines, entries in the list are added when they become active in the current scan line. They are deleted from the list when the last scan lines they are active have been processed.

A scan line array with as many elements as the number of horizontal elements in the current fuzzy array is maintained. Each element of the array contains a depth field and a boolean field called the overlap indicator. The depth field is used to store the depth of the patch whose fuzzy extent surrounds the fuzzy element. The overlap indicator indicates whether the element is under the non-fuzzy region. The depth fields are initialized to negative infinity. The overlap indicators are set to zero.

The pairs of edges in the first active edge list are then processed. The span of each pair of these edges represents the segment of a patch in the current scan line. The overlap indicators of the elements within the span are set to one. The value stored in the depth field of each of these elements is compared with the depth of the patch with respect to the chosen viewpoint. It may be approximated by the maximum depth from the viewpoint to the patch or the patch bounding box. If the former is smaller, it is replaced by the latter.

The computation of the segments of patches requires the finding of the intersections between the current scan line and the active edges. This computation can be avoided by using the leftmost or rightmost position of each edge depending on whether it is at the left or right side of the span. This inaccuracy is allowed because it is always offset by the fuzzy extents of the boundary edges.

The pairs of active edges in the second active edge list are then accessed. The span of each pair represents the segment of the fuzzy extent of a boundary edge in the current scan line. All the overlap indicators of the elements within the span are reset to zero.

After all the records in the active edge lists have been processed, the elements in the scan line array where the overlap indicators are one are within the segment of the non-fuzzy region in the current scan line. The depth fields contain the maximum possible depth of the region in the viewing directions represented by these elements.

The depth stored in each element of the array is compared with the depth field of the corresponding element in the fuzzy buffer. If the latter is larger, it is replaced by the former.

Returning to the Figs., FIG. 8B shows the scan line array after scan-conversion of the active edges of patch A. FIG. 8C shows the scan-conversion of the active edges of patch B. FIG. 8D shows the scan line array after the elements within the fuzzy extents EF_A and EF_B of the boundary edge have been reset.

APPENDIX 4

The Scan-Conversion of the Fuzzy Regions

To obtain the fuzzy regions, a modified scan line algorithm to that of Appendix 3 is used.

An active edge list is maintained during the processing of scan lines. The list contains pairs of vertical edges of the fuzzy extents of patches active in the current scan line. During the processing of successive scan lines, entries in the list are added when they become active in the current scan line. An entry is deleted from the list when the last scan line it is active has been processed. For the arrangement of FIG. 9A (similar to FIG. 8A, but for fuzzy regions), the active edge list is:

E1 E2 E3 E4.

A scan line array with as many elements as the number of horizontal elements in the current fuzzy array is maintained. Each element of the array is a boolean field called the overlap indicator which indicates whether the element is under the fuzzy region.

For each scan line, the entries in the active edge list are accessed. The span between each pair represents the segment of the fuzzy extent of a patch in the current scan line. All the overlap indicators of the elements within the span are set to one.

After the processing of the entries in the active edge list. The array elements whose overlap indicators are one now contain the segment of the fuzzy region in the current scan line.

FIG. 9B shows the state of the scan line array after the elements within the fuzzy extent EF_A of patch A have been updated.

FIG. 9C shows the state of the scan line array after the elements within the fuzzy extent EF_B of patch B have been updated.

APPENDIX 5

Using a Cache to Speed up the Access and Writing of Fuzzy Elements Within the Fuzzy Extent of a Patch

The fuzzy buffer methods described herein require the reading and writing of the fuzzy elements within the fuzzy extent of each patch. Since there are usually substantial overlaps between the fuzzy extents of adjacent patches, the access and update of these elements can be reduced by storing their information in high speed cache.

The cache may be in many forms. One example, shown in FIG. 10, is to use a cache buffer CB and two cross-reference tables CR_1 and CR_2 . The cache buffer CB is a 2D array, in which each element contains fields which are the mirror of the fields on a fuzzy buffer element that need to be read or written. The cross-reference tables CR_1 and CR_2 respectively contain as many elements as the number of rows and columns in the 2D array. Depending on the array it is in, each element in the table contains either a row (CR_1) or a column (CR_2) field which is used to cross-reference the rows and columns of the 2D array with the rows and columns of the current fuzzy array FA. For each element in the cache, the cross-reference tables CR_1 and CR_2 contain a reset switch to indicate whether it has been reset.

Initially, all the reset switches are initialized to one. The centre of the cache buffer CB is mapped to the centre of the fuzzy extent of the first patch to be read. After the fuzzy elements within the fuzzy extent of the first patch are read, information in these elements are stored in the cache buffer CB elements according to the mapping. The mappings of rows and columns in the cache are updated into the cross reference table CR_1 and CR_2 , respectively. The reset switch in the updated elements in these table and the cache buffer CB are set to zero.

Before reading the fuzzy elements within the fuzzy extent of the second patch, data in the cache buffer CB is first read. If an element has already been stored in the cache buffer CB, no access to the cache buffer CB to obtain its information is necessary.

To maximize the efficiency of the cache, any adjacent patch is accessed first. After the processing of a series of patches, there may be a patch whose mapping covers the boundary of the cache. The affected rows or columns are wrapped around. The rows at the opposite edges become the next rows or columns for the patch.

Before the writing of the patch data into the cache buffer CB, each element in the cross-reference table CR_1 and CR_2 within the rows of the columns of the current patch mapping are checked. If the reset switch is on or if the mapping value conforms to the current mapping, the whole row or column of cache elements have not been used and there is no problem in using them. Otherwise, the row or the column needs to be reclaimed. The original data in the whole row or column of cache elements are written back to the fuzzy buffer and then initialized. The affected element in the cross-reference table is updated with the new mapping. After that, the patch data can then be written into these newly-reclaimed cache elements.

The writing from the cache buffer CB to the fuzzy buffer can be performed in burst mode. Hence when such task occurs, several adjacent rows or columns of the cache elements may be written to the fuzzy buffer and re-initialized even though some of the elements may not need to be re-claimed.

APPENDIX 6

Direct Computation of the Fuzzy Region of an Edge

Assume that the equation of the edge is expressed as:

$$x=az+b \tag{1}$$

$$y=cz+d \tag{2}$$

2. Assume that all the coordinates of a viewpoint P in the current viewpoint bounding box is (xO,yO,zO).

3. The image projection of a point T (x,y,z) on the edge with respect to P is

$$X = D \frac{x - xO}{z - zO} \tag{3}$$

$$Y = D \frac{y - yO}{z - zO} \tag{4}$$

4. Since (x,y,z) and (xO,yO,zO) are uncertain, the region comprising all possible values of (X,Y) forms an area which is the fuzzy region of the edge.

5. By substituting (2) into (4) and rearranging,

$$z = \frac{Dd - DyO + Y \ zO}{Y - Dc} \tag{5}$$

If Y is known, the extrema of z can be found by substituting the appropriate extrema of yO and zO into the above equation. All points on the edge having z within these extrema could project at Y from viewpoints in the current viewpoint bounding box.

US 6,172,679 B1

25

6. substituting (1) into (3),

$$X = D \frac{a z + b - x0}{z - z0} \quad (6)$$

Notice that in (6) there is no local extremum of X for changing z. Hence by assigning the extrema of z found from section 5 and the appropriate extreme values of x0 and z0 into (6), the two extrema of X can be found. These values correspond to the maximum and minimum possible X for a particular value of Y.

7. A scan line approach can be used to approximate the fuzzy region of the edge. Substituting (2) into (4) and rearranging,

$$Y = D \frac{c z + d - y0}{z - z0} \quad (7)$$

By assigning the appropriate extreme values of z, y0 and z0 into (7), the maximum and minimum values of Y are obtained. These values are rounded up and down respectively to cover all scan lines the edge may project onto. Each of these scan lines corresponds to a value of Y. From section 5 and 6, the maximum and minimum of X can be found. The area within these values is the scan line segment of the fuzzy region of the edge. The corresponding segment of the expanded fuzzy region of that edge is the further rounding of the segment to fully cover all the grid cells it projects cover.

The above technique computes the expanded fuzzy regions by scan-converting the segments of the regions on the X scan lines. For edges more horizontal than vertical, the expanded fuzzy regions may be more accurately computed by scan-converting the segments of the regions on the Y scan lines.

This is shown in FIG. 14 where the fuzzy region takes on a scan-line shape about the fuzzy region of the edge. This is substantially smaller than the fuzzy extent of the edge.

APPENDIX 7

Scan-Conversion of Patches on the Hemicube Pyramid

This appendix describes an efficient technique for scan-converting a patch on a plane which contains cells with different resolutions. It can be used to compute the form-factor of patches in the radiosity method with or without the use of the fuzzy projection techniques.

The technique uses a data structure called the hemicube pyramid. Each level of the pyramid contains a set of 2D arrays. Each array corresponds to a level and a projection face. Each element of the array corresponds to a pixel on the plane. Note that the hemicube may be a prism rather than a cube and not all its planes need to be active.

An element in the array is called a pyramid element. In addition to the information needed in an original hemicube element, it contains a sub-division indicator and a pointer. The sub-division indicator is initially set to OFF. The indicator becomes active if its level is less than the required accuracy level of patches projected onto it.

The pointer is used to link a pyramid element with one of the pyramid elements at the higher accuracy level and within its region. To enable more accurate sampling, the elements to be linked are usually chosen such that an element accessed by the pointers from an element at the lower accuracy level is as close to the centre of the latter as possible.

26

The organization of the hemicube pyramid corresponding to a face of the hemicube is shown in FIG. 15.

After the initialization of the hemicube pyramid, patches are accessed in decreasing order of accuracy and scan-converted to the hemicube pyramid. This order of access ensures that patches are scan-converted in resolutions equal to or finer than their own optimal projection resolutions and the optimal projection resolutions of patches they might occlude.

For each projecting patch, the hemi-cube projection resolution corresponding to its accuracy level is found. Each hemicube pyramid element belonging to that resolution and under the projection of that patch is accessed.

If the accessed element is already at the most accurate level, or if its sub-division indicator is not on, the ID of the current patch is written into the current element if that patch is closer than the so-far closest patch stored in the element, as in the normal hemicube scan-conversion. If this update occurs, all the ancestors of the current element until the element whose sub-division indicator has already been set to on are accessed. The sub-division indicator of each accessed element is set to on.

Note that the recursive access of ancestor elements can instead be carried out for all elements under the patch projection immediately after they have been found. However, if the possibility that patches are invisible is high, carrying out the recursion after determining that the patch is closer is more efficient as it does not need to be applied on elements if that patch is hidden.

If the sub-division indicator of the current element is ON, patches requiring more accurate form-factor evaluation have been projected onto the element. Therefore, each of its daughter elements under the projection of the current patch is accessed and the above computations are repeated.

After the scan-conversion of all projecting patches, the elements in all the levels are scanned. From the projection information stored in these elements, the form factors of the patches are computed.

If highly accurate form-factor computations and hence very high hemi-cube resolutions are required, relatively large amounts of memory are needed to store the hemi-cube arrays. The initialization and scanning of these arrays for form-factor accumulation is also costly.

Since patches requiring very fine projections are relatively few, the above problem can be overcome by using sparse matrices to store active (sub-divided) elements corresponding to high resolution levels. Each matrix is a one-dimensional array. Active elements are stored in it at positions determined by a hashing function parameterized by pixel positions.

High accuracy form-factor computations can also be achieved using ray-tracing in conjunction with hemicube computations. All the patches are still scan-converted to the hemicube pyramid or the normal hemicube. However, the scan-conversion of patches requiring high accuracy in form-factor computations is only used to determine how other patches are blocked by them. Instead, their form-factors are determined by tracing rays from them.

The content of the hemicube pyramid can be used to speed up the ray-tracing. During the scan-conversion of a patch that is also ray-traced, patches stored in the buffer that are closer to it are determined. When ray-tracing it, these patches may be tested with the rays first as they are more likely to block these rays.

APPENDIX 8

REFERENCES

1. Aggarwal J. K., "Dynamic Scene Analysis," in "Image Sequence Processing and Dynamic Scene Analysis," Huang, T. S. (Eds.) 1981, Springer-Verlag, pp. 40-73.

US 6,172,679 B1

27

2. Appel A. "The Notion of Quantitative Invisibility and the Machine Rendering of Solids", Proc. of the ACM National Conference, 1967, Thompson Books, Washington DC, pp.387-393.
3. Baum, D. R., Rushmeier H. E., Winget J. M., "Improving Radiosity Solutions through the use of Analytically Determined Form-factors", Computer Graphics, Vol 23 Number 3, 1989, pp. 325-334.
4. Bradler, N., Tsotsos, J. K., (Eds) "Motion: Representation and Perception," North-Holland, 1986.
5. Catmull, E. "A Subdivision Algorithm for Computer Display of Curved Surfaces", PhD Thesis, Report UTEC-CSc-74, University of Utah, 1975.
6. Chang, S. S. L., Zadeh, L. A., "On Fuzzy Mapping and Control," IEEE Transactions on Systems, Man, and Cybernetics, Vol.SMC-2, No. 1, January 1972, pp. 30-34.
7. Cohen, M. F., D. P. Greenberg, "The Hemi-Cube: A Radiosity Solution for Complex Environment," SIGGRAPH 85, pg. 31-40.
8. Cohen, M. F., Chen, S. E., Wallace, J. R., Greenberg, D. P., "A Progressive Refinement Approach to Fast Radiosity Image Generation," SIGGRAPH 88, pg. 75-84.
9. Crocker, G. A., "Invisibility Coherence for Faster Scan-Line Hidden Surface Algorithms," SIGGRAPH'84, pp. 315-321.
10. Fournier, A., Fussell, D., "On the power of the Frame Buffer," ACM Transactions of Graphics, April 1988, pp. 103-128.
11. Fuch, H., Kedem, Z., Naylor, B. F., "On Visible Surface Generation by A Priority Tree Structures," SIGGRAPH'80, pp. 124-133.
12. Gordon, D., "Front-to-Back Display of BSP Trees," IEEE CG&A, September 1991, pg. 79-85.
13. Haines, E. A., Wallace, J. R., "Shaft Culling for Efficient Ray-Traced Radiosity," unpublished paper, July 1991.
14. Hornung, C. "A Method for Solving the Visibility Problem", IEEE Computer Graphics and Applications, July 1984, pp.26-33
15. Hubschman, H., Zucker, S. W., "Frame-to-Frame Coherence and the Hidden Surface Computation: Constraints for a Convex World," SIGGRAPH 81, 45-54.
16. Immel, D. S., Cohen, M. F., "A Radiosity Method for Non-Diffuse Environments," Computer Graphics, Vol. 4, 1986, pp. 133-142.
17. Jain, R., Haynes S., "Imprecision in Computer Vision," in "Advances in Fuzzy Sets, Possibility and Applications," Wang, P. (Ed), Plenum Press, 1983, pp. 217-236.
18. Kanatani, K., "Group-Theoretical Methods in Image Understanding," Springer Verlag, 1990.
19. Kaufmann, A., "Theory of Fuzzy Subsets, Vol. 1, Fundamental Theoretical Elements", Academic Press, 1975.
20. Ligomenides, P. A., "Modeling Uncertainty in Human Perception," in "Uncertainty in Knowledge-Based Systems," Bouchon, B., Yager, R. (Eds), Springer Verlag, 1986, pp. 337-346.
21. Lim, H. L., "Fast Hidden Surface Removal Through Structural Analysis and Representation of Objects and Their Contours," CGI'87, Japan, 1987, pp. 75-88.
22. Marks, J., Walsh, R., Christensen, J., Friedell, M., "Image and Intervisibility Coherence in Rendering", Proceedings of Graphics Interface '90, Toronto, Ontario, May 1990, pp. 17-30.
23. Recker, R. J., George D. W., Greenberg D. P., "Acceleration Techniques for Progressive Refinement Radiosity," Proceedings, 1990 Symposium on Interactive 3D Graphics, Snowbird, Utah, Computer Graphics, pg. 59-66.

28

24. Ruspini, E. H., "A New Approach to Clustering," Information Control, Vol. 15, pp. 22-32.
 25. Siedlecki, W., Siedlecka, K., Sklansky, J., "Mapping Techniques for Exploratory Pattern Analysis," in "Pattern Recognition and Artificial Intelligence," Gelsema, E. S., Kanal L. N., (Eds), Elsevier Science, 1988, pp. 277-299.
 26. Sillion, F., Puech, C., "A General Two-Pass Method Integrating Specular and Diffuse Reflection," SIGGRAPH 1989, pg. 335-344.
 27. Subbarao, M., "Interpretation of Visual Motion: A Computational Study," Pitman, London, 1988.
 28. Sutherland, I. E., Sproull, R. F., Schumacker, R. A., "A Characterization of Ten Hidden Surface Algorithms", Computing Surveys, Vol. 6, No. 1, 1974, pp.1-55
 29. Teller, S. J., "Visibility Preprocessing for Interactive Walkthrough," Computer Graphics, Vol. 25, No. 4, July 1991.
 30. Wallace, J. R., Elmquist, K. A., Haines, E. A., "A Ray Tracing Algorithm for Progressive Radiosity", Computer Graphics, Volume 23, Number 3, July 1989, pp. 315-324.
 31. Yee, L., "Spatial Analysis and Planning under Imprecision", North-Holland, 1988.
 32. Zadeh, L. A., "Fuzzy Sets", Information and Control, Vol. 8, 1965, pp. 338-353.
- What is claimed is:
1. A method of reducing the complexity of visibility calculations required for the production of multi-dimensional computer generated images, said method performed on a computer, said method comprising the steps of:
 - prior to an occlusion or invisibility relationship computation (known per se) being carried out on a plurality of surfaces from each viewpoint to be calculated:
 - for selected ones of said surfaces, determining for said viewpoint whether each said selected surface is
 - (a) an always unoccluded surface, an always hidden surface, or a remaining surface; or
 - (b) an always unoccluded surface, or a remaining surface; or
 - (c) an always hidden surface, or a remaining surface;
 - wherein said remaining surface is a surface which is unable to be determined with certainty as to whether it is either unoccluded or hidden;
 - exempting from said occlusion or invisibility relationship computation those surfaces which are either always unoccluded or always hidden;
 - maintaining a record of said remaining surface; and
 - carrying out occlusion or invisibility relationship computations on said remaining surfaces.
 2. A method as claimed in claim 1, wherein the maintaining step comprises creating a previous record.
 3. A method as claimed in claim 1, wherein the maintaining step comprises creating a new record.
 4. A method as claimed in claim 1, wherein said images are selected from the group consisting of graphic images, computer vision data, abstract data and physical data.
 5. A method as claimed in claim 1, wherein the reduction in complexity involves a reduction in the number and/or duration of visibility calculations.
 6. A method as claimed in claim 1, wherein the visibility calculations form part of a series of radiosity calculations.
 7. A method as claimed in claim 6, wherein the exempting step involves the direct computation of said unoccluded surfaces.
 8. A method as claimed in claim 1, wherein one or more of said surfaces are sub-sets of a primary surface.
 9. A method as claimed in claim 1, further including the steps of:

US 6,172,679 B1

29

determining for each boundary of selected ones of said surfaces, the extent to which that boundary defines a boundary area which represents the viewing direction with which the edge of the boundary could be seen;
 determining the projection of said selected ones of said surfaces and determining the projection of the boundary edges of said each selected surface;
 subtracting from the surface projection the boundary edge projection to form an umbra region of said each selected surface; and
 carrying out said occlusion or invisibility relationship computations on said umbra regions and said projection of said each selected surface.

10. A method as claimed in claim **9**, wherein the extent of the boundary area is coincident with a scan-line representation of any of said selected surfaces.

11. A method as claimed in claim **9** or **10**, wherein said calculations are further simplified by detecting, and not carrying out an occlusion or invisibility relationship computation for, surface areas which are totally hidden as a consequence of their depth from the current viewpoint and the presence of nearby surface areas of lesser depth when viewed from the current viewpoint, said detecting of said totally hidden surface areas comprising the steps of:

for each range of viewing directions related to said viewpoint allocating a depth plane to the projection surface of the corresponding surfaces in that viewing direction, each element of the projection surface having a plurality of depth values associated with the viewpoint viewing the corresponding surface area;

for each surface area selecting the depth value for each element and storing same to define the depth values of the projection surface, thus representing the depth value of the umbra region;

similarly determining the depth value of the surface projection; and

comparing the depth value of the umbra region with that of the surface projection, and if the former is smaller, then deeming that surface forming the surface projection is totally hidden.

12. A method as claimed in claim **11** wherein the selecting step selects the smallest of the approximate largest depth values of surfaces projected on each element.

13. A method as claimed in claim **11** wherein the determining step selects the smallest possible depth value of each element.

14. A method as claimed in claim **1**, wherein said calculations are further simplified by detecting, and not carrying out an occlusion or invisibility relationship computation for, surface areas which are totally unoccluded, said detecting of said totally unoccluded areas comprising the steps of:

prior to the carrying-out step, creating and storing a projection for each of selected ones of said surfaces or surface elements and for each element of each stored projection, indicating whether the element is either:

- (a) not within the projection of any of said selected surfaces;
- (b) within the projection of a single one of said selected surfaces; or
- (c) within the projection of more than one of said selected surfaces; and

scanning the projection of each said selected surface and where, in all elements for one projection are indicated by (b) in step creating, deeming the surface so projected is totally unoccluded.

15. A method as claimed in claim **1**, wherein said surfaces are arranged in a hierarchy represented by varying levels of details.

30

16. A method of reducing the visibility related computations in an environment consisting of three-dimensional abstract or physical surfaces, said method comprising the steps of:

prior to a visibility computation being carried out, defining one or more projection surfaces (known per se) for the purpose of generating projections of surfaces or surface elements with respect to a current viewpoint or its possible occurrences;

for selected surfaces, determining whether those surfaces or their surface elements are always invisible to said viewpoint by computing and comparing their projection mappings on said projection surface or surfaces;

ignoring or treating specially some or all surfaces or surface elements which are always invisible to said viewpoint during the actual visibility or visibility related computations;

whereby the visibility related computations are reduced.

17. A method of reducing the visibility, radiosity or visibility related computations in an environment consisting of three-dimensional abstract or physical surfaces, said method comprising the steps of:

prior to a visibility computation being carried out, defining one or more projection surfaces for the purpose of the projection of surfaces or surface elements with respect to a current viewpoint;

dividing each of said projection surfaces into at least one regular or irregular grids;

defining a data structure which organizes computer storage for storing of projections on said grid;

for each of the selected surfaces or surface elements, projecting said surfaces or their surface elements onto the projection surfaces, computing grid cells which are under the projections;

ignoring or treating specially surface elements which are always visible to said viewpoint during the actual visibility or visibility related computations for said viewpoint;

whereby the visibility, radiosity or visibility related computations are reduced.

18. A method of storing computer generated image data, said method performed on a computer having a physical data store, said method comprising the steps of

identifying a plurality image portions visible to a viewpoint;

selecting those portions having a predetermined image characteristic and storing same in an organization storage structure;

for a first one of said selected portions, allocating a portion identifier and storing same together with image data of said first portion;

for the remaining said selected portions, calculating a difference value identifier and storing same together corresponding image data of said selected portion, wherein any said selected portion is addressable by means of said portion identifier and one or more said difference value identifiers.

19. A method as claimed in claim **18** wherein the calculated difference value identifier represents an address difference value between a said remaining selected portion and said first selected portion.

20. A method as claimed in claim **18** wherein the calculated difference value identifier represents an address difference value between a said remaining selected portion and another said remaining selected portion, in which one of said

US 6,172,679 B1

31

remaining selected portions has a calculated difference value identifier that represents an address difference value between a said remaining selected portion and said first selected portion.

21. A method as claimed in claim 18 wherein said storage structure is non-contiguous. 5

22. A method as claimed in claim 21 wherein said storage structure is a linked-list.

23. A method as claimed in claim 18 wherein said storage structure is contiguous. 10

24. A method as claimed in claim 21 wherein said storage structure is an array.

25. A method as claimed in claim 18 wherein said image portions are stored based upon a plurality of said predetermined image characteristics to give multiples sets of image data, in which any element of any set can be manipulated based upon a viewing from another viewpoint. 15

26. A method as claimed in claim 25 wherein said manipulation involves adding an element to a set, deleting an element from a set, or copying an element between any two or more sets. 20

27. A method as claimed in claim 26 wherein a plurality said predetermined image characteristics are available and only the smallest of the plurality less one corresponding set(s) are(is) retained, the other one being redundant. 25

28. A method of reducing the complexity of calculations required for the production of multi-dimensional computer generated images or the reduction of multi-dimensional data to multi-dimensional data having at least one less dimension, said method performed on a computer, said method comprising the steps of: 30

observing an object surface with a current viewpoint, and defining a projection surface for said viewpoint;

creating a working surface and mapping each of said projection surfaces onto said working surface; 35

computing a region that contains all the mappings;

whereby the complexity of calculations required for the production of multi-dimensional computer generated images or the reduction of multi-dimensional data to multi-dimensional data having at least one less dimension is reduced. 40

29. A method as claimed in claim 11 wherein said calculations for said remaining surfaces are further reduced in complexity by: 45

establishing a plurality of accuracy levels required to be reached by form-factor computations of said remaining surfaces;

allocating one of said accuracy levels to each of said always occluded surfaces, to each of said always hidden surfaces, and to each element of said projection; 50

updating the accuracy level of each said surface with the accuracy level of the corresponding element of the

32

projection if the latter is larger, and subsequently scan-converting each of said surfaces in order commencing with those having the highest accuracy level and ending with those having the lowest accuracy level, the updated accuracy levels indicating the maximum level of accuracy for form-factor computations for each said surface, and determining if the updated accuracy level for each said surface is higher than that which said surface can provide, and if so, recursively subdividing the corresponding said surface into a plurality of sub-surfaces until the corresponding accuracy levels of said sub-surfaces match that of said surface.

30. A method as claimed in claim 29 wherein said form-factor computations are performed using a ray-tracing technique or a hemi-cube technique.

31. A method of scan-converting a plurality of surfaces of a multi-dimensional computer generated image, said method performed on a computer, said method comprising the steps of: 5

creating a data structure comprising a plurality of two-dimensional arrays, each said array relating to a corresponding accuracy level for a projection of said surfaces as viewed from a viewpoint, the elements of each array being pixels on a projection plane;

for each said element, providing an associated sub-division indicator, the sub-division indicator being adapted to indicate an active sub-division of the corresponding said element;

initializing each said sub-division indicator to OFF;

for each patch of each said surface, defining a desired accuracy for same and scan-converting the projection including said patches in decreasing order of accuracy into corresponding elements of said structure, and setting the sub-division indicator for each said element of the highest accuracy level to ON if that element corresponds to one of said patches;

for each said element of the surface, if the sub-division indicator is ON, accessing the sub-elements of said element which are under the projection, and if the indicator is not ON, updating the patch in form to said sub-element, if the depth of the patch is less than the depth stored in the sub-element; and

accessing each said element and adding a form-factor contribution of the element to the form-factor of a record of the patch whose own identifier is stored in the patch.

32. A method as claimed in claim 17, where the data structures are either at least one z-buffer or at least one quadtree.

* * * * *